Click to prove you're human

Conditional randomization test large language model

Artificial intelligence has been advancing at lightning speed, and Large Language Models (LLMs) are at the forefront of this revolution. However, with rapid innovation comes the critical need to thoroughly test and validate these models to ensure they're reliable, relevant, and actually beneficial for your business. In this guide, we'll walk through the ins and outs of AI model evaluation—from automated benchmarking to human evaluation—and show you how tools like LangSmith can streamline the process. Whether you're just starting out or looking to refine your current AI setup, our goal is to help you test and validate Large Language Models more effectively. Why AI Model Evaluation Matters Simply put, AI model evaluation is about making sure your Large Language Model does what you need it to do in the real world. This involves checking: Consistency: Do the model's outputs stay high-quality across different scenarios? Relevance: Are the responses actually meaningful and valuable for your industry or niche? Reliability: Can it handle unexpected inputs without breaking or producing nonsense? Answering these questions gives you a clear sense of how well your LLM is performing—and most importantly, whether it's driving results for your organization. Different Ways to Evaluate LLMs 1. Automated Benchmarking Programmatic Testing: Compare your model's outputs against a known baseline to measure things like accuracy or sentiment. Performance Metrics: Use clear, repeatable statistics (like LLM performance metrics) to see how one version stacks up against another. Scalability: Evaluate large data sets quickly and consistently. Think of automated benchmarking as your first checkpoint. It's a cost-effective way to spot big issues early on, so you can make improvements fast. 2. Human Expert Evaluation Domain Expertise: Having people who truly know your industry evaluate the AI's responses. Real-World Context: Experts can tell if an answer is genuinely practical or just "sounds good." Subtle Insights: Humans catch nuances—cultural, ethical, or otherwise—that might fly under the radar of automated methods. For tasks where context matters a lot, human evaluation is essential. It's often the difference between an AI that looks good on paper and one that truly works in practice. 3. LLM-as-Judge Evaluation AI-Based Assessment: Use one LLM to evaluate another LLM's outputs. Fast, High-Volume Feedback: This method can handle a ton of data quickly, making it ideal for large-scale testing. Focused Testing: Check specific things like factual correctness or tone consistency. Approaches like LLM-as-Judge—and frameworks like RAGAS—are getting more popular because they offer rapid feedback loops that help fine-tune your model according to your exact business needs. The SMART Evaluation Framework A little structure goes a long way. That's where SMART comes in: Specific: Make sure your evaluation criteria directly relate to your business goals. Measurable: Identify quantifiable indicators—like accuracy or user satisfaction—to gauge performance. Actionable: Translate your findings into real steps your developers can take. Relevant: Keep your eyes on what matters to both your stakeholders and the market. Time-Bound: Schedule regular check-ins so you're always improving. By applying this SMART approach, you'll keep your Large Language Model testing on track and aligned with your evolving objectives. Spotlight on LangSmith If you're looking for an all-in-one tool to help with testing and validating Large Language Models, LangSmith is worth a closer look. It offers: End-to-End Monitoring: Keep an eye on your LLM performance metrics in real time. Structured Frameworks: Ready-to-use templates for testing, making your process smoother. Data Management: Easily store, label, and organize large sets of data for better AI model evaluation. Automated Testing: Slash manual work and speed up deployment with built-in automation features. Seamless Integration: Plug LangSmith into your existing workflow without missing a beat. As AI evolves, tools like LangSmith are built to adapt right alongside it, giving you a future-proof way to manage and optimize your models. Practical Tips for Implementation 1. Define Clear Objectives Pinpoint Use Cases: Know exactly what problems your model is solving. Establish Metrics for Success: Decide if you care most about user satisfaction, raw accuracy, or another key metric. Set Thresholds: Figure out what's acceptable performance so you know when to iterate or move on. 2. Combine Multiple Evaluation Methods Mix Automated and Human Approaches: Get a balanced view by pairing automated benchmarking with human evaluation. Ongoing Monitoring: Keep track of your model's performance over time—issues can crop up unexpectedly. Diverse Feedback: Gather insights from domain experts, end-users, and anyone else who interacts with the model. 3. Keep Standards Consistent Document Everything: Record your evaluation protocols so you can replicate tests or train new team members. Refine Your Criteria: As your business needs change, update your LLM-as-Judge or RAGAS criteria accordingly. Train Your Team: Make sure everyone involved understands SMART principles and how to use tools like LangSmith. 4. Take Action on What You Find & Experiment! Create Feedback Loops: Send test results back to developers or data scientists promptly. Prioritize Changes: Focus on the updates that give you the biggest improvement first. Monitor Trends: Keep a historical log of your LLM performance metrics to watch how the model evolves. Looking Ahead As AI continues to advance, testing and validating Large Language Models will likely become even more integrated into everyday development. Look out for: Ready to Take the Next Step? Whether you're taking your first stab at testing and validating Large Language Models or want to polish an existing approach, a strong evaluation process can make all the difference. At Compoze Labs, we specialize in creating and implementing AI model evaluation frameworks tailored to your goals. View PDF Abstract:Pre-trained large language models (PLMs) underlie most new developments in natural language processing. They have shifted the field from application-specific model pipelines to a single model that is adapted to a wide range of tasks. Autoregressive PLMs like GPT-3 or PaLM, alongside techniques like few-shot learning, have additionally shifted the output modality to generation instead of classification or regression. Despite their ubiquitous use, the generation quality of language models is rarely evaluated when these models are introduced. Additionally, it is unclear how existing generation tasks--while they can be used to compare systems at a high level--relate to the real world use cases for which people have been adopting them. In this work, we discuss how to adapt existing application-specific generation benchmarks to PLMs and provide an in-depth, empirical study of the limitations and capabilities of PLMs in natural language generation tasks along dimensions such as scale, architecture, input and output language. Our results show that PLMs differ in their applicability to different data regimes and their generalization to multiple languages and inform which PLMs to use for a given generation task setup. We share best practices to be taken into consideration when benchmarking generation capabilities during the development of upcoming PLMs. From: Joshua Maynez [view email] [v1] Thu, 29 Jun 2023 08:59:40 UTC (895 KB) In the rapidly evolving landscape of artificial intelligence, understanding the nuances of model evaluation is crucial, especially when it comes to large language models (LLMs). One of the emerging methodologies gaining traction is the Conditional Randomization Test (CRT), a statistical technique designed to assess the performance of these complex models in a more robust manner. Unlike traditional evaluation methods, CRT offers a way to control for confounding variables and better understand the true impact of various factors on model outputs. In this blog post, we'll delve into the mechanics of the Conditional Randomization Test, explore its significance in the context of large language models, and highlight how it can enhance our understanding of model behavior and reliability. Large Language Models, Explained With A Minimum Of Math And Jargon Large language models are advanced AI systems designed to understand and generate human-like text. They work by analyzing vast amounts of text data to learn patterns in language, allowing them to predict what words or phrases are likely to come next in a sentence. Imagine having a conversation with a friend who has read every book and article ever written; they can provide insightful responses based on that knowledge. In the context of conditional randomization tests, these models can help researchers assess how different factors influence outcomes in text generation, ensuring that the results are not just random but meaningful. By using these tests, we can better understand the underlying mechanisms of language generation and improve the reliability of AI-generated content. www.understandingai.org Figure 1 From A Conditional Randomization Test For Sparse Logistic In Figure 1 of our exploration of conditional randomization tests for sparse logistic models, we visually illustrate the core concept of how these tests function within the framework of large language models. The figure presents a flowchart that outlines the steps involved in conducting a conditional randomization test, highlighting the process of isolating the effects of specific variables while controlling for confounding factors. By employing this method, researchers can more accurately assess the significance of variable interactions in sparse logistic regression, ultimately leading to more reliable interpretations of model outcomes. This foundational understanding sets the stage for deeper discussions on the implications and applications of conditional randomization tests in the realm of large language models. www.semanticscholar.org (pdf) Conditional Randomization Test Of Heterogeneus Effect On In the realm of statistical analysis and machine learning, the conditional randomization test (CRT) plays a pivotal role in assessing heterogeneous effects, particularly when evaluating the performance of large language models (LLMs). This method allows researchers to determine whether the effects of certain variables differ across various subgroups within the data. By implementing a CRT, one can effectively control for confounding factors and ensure that the observed effects are not merely due to chance. In the context of LLMs, this approach is vital for understanding how different demographic or contextual factors influence model outcomes, enabling developers to create more equitable and robust language processing systems. The use of a PDF format for sharing CRT methodologies and results not only enhances accessibility but also fosters collaborative research efforts, allowing practitioners to replicate findings and build upon existing knowledge in the field. www.researchgate.net The Unconditional And Conditional Performance Of Our Conditional In the realm of Conditional Randomization Tests applied to Large Language Models (LLMs), understanding the concepts of unconditional and conditional performance is crucial. Unconditional performance refers to the overall effectiveness of a model without any specific constraints or conditions applied to the data input. This metric provides a baseline for evaluating how well the model operates in general scenarios. In contrast, conditional performance assesses the model's capabilities under specific circumstances or conditions, such as particular input types or contextual factors. By examining both unconditional and conditional performance, researchers can gain deeper insights into the model's robustness and adaptability, ultimately enhancing its application in various tasks and improving its predictive accuracy. www.researchgate.net Do Large Language Models Truly Understand What They're Saying? You Might Also Like: Oil Loving Skincare Secret To Glowing Large language models (LLMs), like those used in natural language processing, have garnered significant attention for their impressive ability to generate human-like text. However, a critical question arises: do these models truly understand what they are saying? While LLMs can produce coherent and contextually relevant responses, their "understanding" is fundamentally different from human comprehension. They rely on patterns learned from vast amounts of data rather than genuine semantic awareness. This distinction is crucial when evaluating their outputs, as LLMs can sometimes generate misleading or nonsensical information despite appearing articulate. The Conditional Randomization Test offers a framework for assessing the extent of this understanding by evaluating the models' responses in various contexts, thereby shedding light on their capabilities and limitations. www.masaischool.com The model-X conditional randomization test is a generic framework for conditional independence testing, unlocking new possibilities to discover features that are conditionally associated with a response of interest while controlling type I error rates. An appealing advantage of this test is that it can work with any machine learning model to design powerful test statistic. In turn, the common practice in the model-X literature is to form a test statistic using machine learning models, trained to maximize predictive accuracy with the hope to attain a test with good power. However, the ideal goal here is to drive the model (during training) to maximize the power of the test, not merely the predictive accuracy. In this paper, we bridge this gap by introducing novel model-fitting schemes that are designed to explicitly improve the power of model-X tests. This is done by introducing a new cost function that aims at maximizing the test statistic used to measure violations of conditional independence. Using synthetic and real data sets, we demonstrate that the combination of our proposed loss function with various base predictive models (lasso, elastic net, and deep neural networks) consistently increases the number of correct discoveries obtained, while maintaining type I error rates under control. Article Open access 02 August 2021 Article Open access 29 April 2023 Discover the latest articles and news from researchers in related subjects, suggested using machine learning. Use our pre-submission checklist Avoid common mistakes on your manuscript. An important task in modern data analysis is identifying a subset of explanatory features from a larger pool that are associated with a response of interest. For instance, suppose a nutritionist is interested in identifying which nutrient intakes are associated with body mass index. This information, in turn, could be leveraged to develop diets that improve human health (Chen & Li, 2013). Another example is a geneticist who is interested in finding which genetic mutations of a virus have developed resistance to a specific antiviral drug. To quote Rhee et al. (2006): "[This understanding] is essential for designing new antiviral drugs and for using genotypic drug resistance testing to select optimal therapy." In such applications, there are often a large number of explanatory features with complex dependencies and a limited number of observations, which make the selection problem a particularly difficult task (Shah & Peters, 2020). To stress this point, consider the genetic example from above and notice that a feature selection procedure can make erroneous selections of two kinds: (i) false positives—it may select mutations that do not have any effect on drug resistance; and (ii) false negatives—it may fail to discover genetic mutations that in fact developed resistance to the antiviral drug. This paper presents a novel cost function and learning scheme whose goal is to increase the number of discoveries reported by statistical methods for feature selection, while maintaining the rate of false positives under control.To formalize the problem, we denote by $Y \in \mathbb{R}$ the response variable and by $X = (X_1,X_2,\dots ,X_d) \in \mathbb{R}^d$ the feature vector. For example, imagine that Y is a measure of drug resistance, and $X_j \in \mathbb{R}$—the jth entry in the vector X—indicates the presence or absence of a genetic mutation in a specific location. Conditional independence testing (Candès et al., 2018; Tansey et al., 2021; Liu et al., 2020; Zhang et al., 2012) deals with the question of whether a specific feature X_j is independent of the response variable $Y\in \mathbb{R}$ after accounting for the effect of all the other features $X_{-j} = (X_1,\dots ,X_{j-1}, X_{j+1},\dots ,X_d) \in \mathbb{R}^{d-1}$. That is, the null hypothesis we seek to test is$$\begin{aligned} H_{0,j}: X_j \perp !\!\! \perp Y \mid X_{-j} \end{aligned}$$against the alternative $H_{1,j}: X_j ot \!\perp \!\!\! \perp Y \mid X_{-j}$. In plain words, $H_{0,j}$ inquires whether the knowledge of X_j adds additional information about the response Y beyond what is already contained in X_{-j}. Consequently, we say that the jth feature is null (i.e., unimportant) if $H_{0,j}$ is true, and non-null (i.e., important) if $H_{0,j}$ is false.Imagine we are given n data points $\{(X^i,Y^i)\}_{i=1}^n$, where $X^i \in \mathbb{R}^d$ and $Y^i\in \mathbb{R}$ are drawn i.i.d. from a joint distribution P_{XY}. Statistical tests for conditional independence leverage the observed data to formulate a decision rule on whether to reject $H_{0,j}$ and report that the feature X_j is likely to be important, or accept $H_{0,j}$ in cases where there is not enough evidence to reject the null. A valid test for conditional independence should control the probability of making type I error (i.e., rejecting $H_{0,j}$ when it is in fact true), by returning a p-value, a random variable that is super-uniform under the null. This p-value is then used to rigorously control the type I error rate at any user-specified level α, e.g., of 5%. The power of the test (higher is better) is the true positive rate, being the probability of correctly reporting that the non-null X_j is indeed important.Thus far we formulated the problem of testing for a specific feature, however, in many scientific applications (e.g., genome wide association studies Sesia et al., 2020b; Benner et al., 2016) we are interested in selecting a subset of features that are associated with the response. In other words, we wish to test for all $H_{0,j},\) \(j=1,\dots d\) simultaneously while controlling some statistical notion of type I error. To this end, imagine we have at hand d p-values, one for each $H_{0,j}$, which we denote by $p_j\), \(j=1, \dots ,d\). Recall that each p-value allows us to control the type I error for a specific hypothesis at a desired rate \(\alpha\). However, when testing for all the hypotheses simultaneously we must account for the multiplicity of the tests, as otherwise the probability that some of the true null hypotheses are rejected by chance alone may be excessively high; see the survey by Bender and Lange (2001) for details. Hence, we follow the seminal work by Benjamini and Hochberg (1995) and define our task as follows: identify the largest possible subset $\hat{\mathcal {S}}\subseteq \{1,\dots ,d\}$ of non-null features1 while ensuring that the false discovery rate (FDR), defined as$$\begin{aligned} \text {FDR} = \mathbb {E}\left[\frac{|\hat{\mathcal {S}} \cap \mathcal {H}_0 |}{\text {max}\{|\hat{\mathcal {S}}|,1\}}\right] , \end{aligned}$$falls below a nominal level of choice $q\in (0,1)$, e.g., $q=0.1$ or $q=0.2$. Above, $\mathcal {H}_0 \subseteq \{1,\dots ,d\} {\setminus } \mathcal {H}_1$ contains the indices of the null features for which $H_{0,j}$ is true, $\mathcal {H}_1$ is the set of non-null features, and $|\cdot |$ returns the set-size. In words, the FDR is the expected proportion of true nulls among the rejected hypotheses. In our genetic example, the ability to control the FDR allows us to form a list of discoveries such that the majority of the selected mutations are expected to be conditionally associated with drug resistance, on average. This information may be valuable when allocating costly resources, required to further study the implications of the reported discoveries (Hawinkel et al., 2019; Manolio et al., 2009). The power of the selection procedure (larger is better) is defined as $\mathbb {E}\left[{| \hat{\mathcal {S}} \cap \mathcal {H}_1 |} / {|\mathcal {H}_1|}\right]$, being the expected proportion of non-nulls that are correctly selected among all the true non-nulls. Controlling the FDR can be achieved by plugging the list of p-values $p_j,\) \(j=1,\dots , d\) into the Benjamini-Hochberg procedure (BH) (Benjamini & Hochberg, 1995), as described in Sect. 2.2. Importantly, the power of this feature selection procedure is affected by the ability of the underlying conditional independence test to generate small p-values for the non-null features, indicating strong evidence against the corresponding null hypotheses. Our goal in this work is to increase the power of the above controlled feature selection pipeline by forming data-driven test statistics that powerfully separate non-nulls from nulls.The model-X conditional randomization test (Candès et al., 2018) and its computationally efficient version—the holdout randomization test (HRT) (Tansey et al., 2021) that we study in this paper—are methods for conditional independence testing that gain a lot of attention in recent years. These tools are very attractive since they can leverage any powerful predictive model to rigorously test for $H_{0,j}$, under the assumption that the distribution of the explanatory features P_X is known. As such, model-X tests are especially beneficial in situations where copious amount of unlabeled data is available compared to labeled data, e.g., as happen in genome wide association studies (Sesia et al., 2020a, b; Bates et al., 2020). At a high level, the HRT is carried out by first splitting the data into a training set and a test set, and fit an arbitrary predictive model on the training set. Next, a dummy copy $\tilde{X}_j$ of the original feature X_j is sampled for each sample in the test set, such that $\tilde{X}_j$ is independent of Y given X_{-j} by construction. In our running example, one can think of $\tilde{X}_j$ as a fake genetic mutation (sampled by a computer program) that does not carry any new information about Y if we already know all the other mutations. The test proceeds by comparing, for each observation in the test set, the accuracy of the trained predictive model applied to (i) the original feature vector X, and (ii) its modified version for which only X_j is replaced by $\tilde{X}_j$. Here, a decrease in the overall accuracy can serve as an evidence that X_j is associated with Y after accounting for the effect of X_{-j}.In practice, the HRT is deployed by treating the machine learning model as a black-box, fitted to maximize predictive accuracy with the hope to attain good power. However, the ultimate objective here is to maximize the power of the test and not merely the model's accuracy, and, currently, there is no method to balance between the two. To stress this point further, there is a growing evidence that modern machine learning algorithms tend to excessively rely on spurious features—null features that are solely correlated with the non-null ones—to make accurate predictions (Sagawa et al., 2019; McCoy et al., 2019; Arjovsky et al., 2019; Peters et al., 2015; Khani & Liang, 2021). Loosely speaking, in such cases, the model tends to up-weight the importance of null features at the cost of down weighting the importance of the non-nulls. The consequence of this undesired phenomenon is that the predictive model may only show a mild drop in accuracy when replacing an important feature with its dummy copy. This renders the test to lose power, as the test accounts for the effect of the spurious correlations by design.1.1 Our contributionWe propose novel learning schemes to fit models that are designed to explicitly increase the power of the HRT. This is done by 'looking ahead' (during training) to what will happen when the predictive model is integrated within the HRT procedure. This stands in striking contrast with the common practice, in which the model is merely trained to maximize accuracy to the extent possible. Concretely, the core of our contribution is the formulation of a novel loss function that encourages the model's accuracy to drop when replacing the original feature X_j with the dummy one $\tilde{X}_j$, seeking maximum risk discrepancy (MRD). We introduce in Sect. 3 a general, theoretically motivated, stochastic optimization procedure that enables us to augment our MRD loss to existing cost functions used in the variable selection literature, e.g., sparsity promoting penalty for deep neural networks. That section also includes a specialized learning scheme to fit sparse linear regression models (such as lasso) with our proposed loss. Extensive experiments on synthetic and real data sets are presented in Sect. 5, showing that the combination of the proposed MRD approach with existing regression algorithms consistently improves the power of the underlying controlled feature selection procedure. In particular, in cases where the signal is weak (i.e., in low power regimes), our method may even achieve more than 90% relative improvement compared to existing methods. Importantly, the MRD scheme preserves the validity guarantee of the p-values generated by HRT. Furthermore, we demonstrate that our new proposal is fairly robust to situations where the distribution of P_X—required to sample the dummy features—is estimated from data. These situations are of course ubiquitous in real-world applications. Lastly, a software package that implements our methods, with code for reproducing experiments, is attached to the Supplementary Material.The model-X randomization test provides a valid p-value for $H_{0,j}$ in finite samples without making modeling assumptions on the conditional distribution of $Y \mid X$. However, this approach assumes that the conditional distribution of $X_j \mid X_{-j}$ is known. The original conditional randomization test (CRT), developed by Candès et al. (2018), requires fitting a regression model many times in order to be carried out precisely. The HRT, which is extensively used in this paper, bypasses this computational issue at the expense of data splitting. Specifically, this test starts by dividing the data into disjoint training and testing sets. The training set is used to fit an arbitrary regression model $\hat{f}$, formulating a test statistic function $T(X_j, X_{-j},Y;\hat{f}) \in \mathbb {R}$, e.g., the model's prediction error. Let $\{(X_i,Y_i)\}_{i\in \mathcal {I}}$ be the samples of the testing set. The HRT proceeds by repeating the following two steps for each $k=1,\dots ,K$ (treating $\hat{f}$ as a fixed function): Sample a dummy feature $\tilde{X}^i_j \sim P_{X_j|X_{-j}}(X_j^i|X_{-j}^i)$, for all samples in the test set $i\in \mathcal {I}$. Compute the test statistic using the dummy variables $$\begin{aligned} \tilde{t}_j^{(k)} \leftarrow \frac{1}{|\mathcal {I}|} \sum _{i\in \mathcal {I}} T(\tilde{X}_{j}^i,X_{-j}^i,Y^i;\hat{f}). \end{aligned}$$ Finally, a p-value $\hat{p}_j$ for $H_{0,j}$ is constructed by computing the empirical quantile level of the true test statistic $t^* \leftarrow \frac{1}{|\mathcal {I}|} \sum _{i\in \mathcal {I}} T(X_j^i, X_{-j}^i,Y^i;\hat{f})$ among the dummy statistics $\tilde{t}_j^{(1)},\dots ,\tilde{t}_j^{(K)}$:$$\begin{aligned} \hat{p}_j=\frac{1+ \left| \{ k : t^* \ge \tilde{t}_j^{(k)} \}\right| }{{K+1}}. \end{aligned}$$Importantly, the p-value $\hat{p}_j$ in (1) is valid, i.e., $\mathbb {P}[\hat{p}_j \le \alpha \mid H_{0,j} \text {is true}] \le \alpha$. This is because (i) $\tilde{X}_j \perp !\!\! \perp Y \mid X_{-j}$ by construction, and (ii) under the null, the random variables $t^*, \tilde{t}_j^{(1)}, \dots , \tilde{t}_j^{(K)}$ are exchangeable (Candès et al., 2018).The validity of the p-value holds true for any choice of predictive model $\hat{f}$, and for any choice of test statistic $T(\cdot ,\cdot)$ used to compare the distributions of $(Y, {X}_j, X_{-j})$ and $(Y, \tilde{X}_j, X_{-j})$. Katsevich and Ramdas (2020) proved that the likelihood ratio is the uniformly most powerful statistic, however, it requires the knowledge of the true conditional distribution of $Y \mid X$, which is unknown. In practice, a common choice for a test statistic—which we use in our proposal as well—is the squared error (Tansey et al., 2021; Bates et al., 2020) for which$$\begin{aligned} T(X_j, X_{-j},Y;f)=(Y-\hat{f}(X_j, X_{-j}))^2, \end{aligned}$$where we use the notation $\hat{f}(X_j, X_{-j}) = \hat{f}(X)$ to stress that $\hat{f}$ is applied on the vector X with its original jth feature X_j, whereas $\hat{f}(\tilde{X}_j, X_{-j})$ is applied to the same feature vector except that the original X_j is replaced by its dummy copy $\tilde{X}_j$. To handle the controlled feature selection problem, the HRT p-values p_j, $j=1,\dots ,d$ are most often plugged into the Benjamini-Hochberg procedure (BH) (Benjamini & Hochberg, 1995) procedure, as discussed next.2.2 FDR control via the BH procedureArmed with a list of d p-values p_j, one for each $H_{0,j}$, we can apply the BH procedure (Benjamini & Hochberg, 1995) to control the FDR. This method operates as follows. First, sort the given d p-values in a non-decreasing order; we denote by $p_{(k)}$ and $H_{0,(k)}$ the kth smallest p-value and its corresponding null-hypothesis, respectively. Then, for a given FDR target level q, find the largest k_0 such that $p_{(k_0)} \le \frac{k_0}{d}q$. Lastly, reject the null hypotheses $H_{0,(1)}, H_{0,(2)}, \dots , H_{0,(k_0)}$. Importantly, the procedure described above is guaranteed to control the FDR under the assumption that the p-values are independent. While the p-values we construct in this work do not necessarily satisfy the independence assumption, it is known that the BH procedure often controls the FDR empirically even when the p-values are dependent (except adversarial cases). This empirical observation is also endorsed in our experiments, standing in line with many other methods that plug the dependent p-values generated by CRT/HRT to BH (Tansey et al., 2021; Candès et al., 2018). With that said, it is worth noting that there exists a variant of the BH that provably controls the FDR for any arbitrary dependency structure of p_j, $j=1,\dots ,d$, however, this comes at the cost of reduced power (Benjamini & Yekutieli, 2001).In this section, we introduce our MRD approach to fit predictive models with the goal to explicitly improve the power of the model-X tests. We begin by describing the set of ideas that construct the theoretical motivation. Then, we present a general learning scheme to fit an arbitrary predictive model (e.g., neural network) using the MRD loss (4). Lastly, we design a specialized learning scheme for sparse linear regression models (e.g., lasso).3.1 Main ideaA central definition in our proposal is that of the risk discrepancy, expressed as$$\begin{aligned} \text {RD}_j = \mathbb {E}[T(Y,\tilde{X}_j,X_{-j}; \hat{f}) - T(Y,X_j,X_{-j}; \hat{f})], \end{aligned}$$where $\hat{f}$ is a fixed (pre-trained) predictive model, and $T(\cdot ,\cdot)$ is the squared error (2). In the context of the HRT (Sect. 2.1), a test with good power would have large and positive empirical risk discrepancy for the non-null features $\{j \in \mathcal {H}_1\}$. Therefore, during training, we wish to encourage the model $\hat{f}$ to maximize $\text {RD}_j$ only for that group of features. While this is infeasible because we do not have apriori knowledge of which of the features are in $\mathcal {H}_1$, we do have access to the sampling distribution of the conditional null through the sampling of the dummy copies $\tilde{X}_j$. This property is formally presented in the following well-known result in the model-X literature (Candès et al., 2018), which we will use later to formulate our loss. Proposition 1 (Candès et al., 2018) Take $(X,Y) \sim P_{XY}$, and let $\tilde{X}_j$ be drawn independently from $P_{X_j \mid X_{-j}}$ without looking at Y. If $Y \perp !\!\! \perp X_j \mid X_{-j}$, then $(Y,\tilde{X}_j,X_{-j}) \overset{d}{=} (Y, {X}_j,X_{-j})$. Above, the notation $\overset{d}{=}$ stands for equality in distribution. For completeness, the proof of the above statement is given in Supplementary Section A. A consequence of Proposition 1, important to our purposes, is given in the next corollary; see also König et al. (2021). Corollary 1 Let $\hat{f}$ be a fixed predictive model, and $T(\cdot ; \hat{f})$ be any test statistic. Under the setting of Proposition 1, if $(Y,\tilde{X}_j,X_{-j}) \overset{d}{=} (Y,{X}_j,X_{-j})$, then $\mathbb {E}[T(Y,\tilde{X}_j,X_{-j}; \hat{f}) - T(Y,X_j,X_{-j}; \hat{f})]=0$. This motivates our formulation of a training procedure that drives the model to a solution in which the empirical risk discrepancy (evaluated on the training data) is maximized for all features, knowing that this quantity is guaranteed to be small for the null features at test time. In fact, in Proposition 2, which is presented formally in Sect. 3.3, we prove that for a linear model with uncorrelated features, the solution that minimizes the population version of our objective function (described next) satisfies that the estimated regression coefficients of the null variables are guaranteed to be small for the null features at test time.3.2 A general learning schemeWe now formalize the idea presented above. Given a training set of m i.i.d samples $\{(X^i,Y^i)\}_{i=1}^m$, our procedure begins with a generation of a dummy copy $\tilde{X}^i_j$ for each sample and feature:$$\begin{aligned} \tilde{X}^i_j \sim P_{X_j \mid X_{-j}}(X^i_j \mid X^i_{-j}), \quad i=1,\dots ,m, \ \ j = 1 \dots d. \end{aligned}$$Denote the collection of training features by $\{\textbf {X}} \in \mathbb {R}^{m \times d}$, where X^i is the ith row in that matrix, and by $\{\textbf {Y}} \in \mathbb {R}^{m}$ a vector that contains the response variables Y^i as its entries. Let $\mathcal {J}_{\text {base}}({\textbf {Y}},{\textbf {X}}; f)$ be the objective function of the base predictive model, expressed as$$\begin{aligned} \mathcal {J}_{\text {base}}({\textbf {Y}},{\textbf {X}}; f) = \frac{1}{m} \sum _{i=1}^m \ell (f(X^i),Y^i) + \alpha \mathcal {R}(f). \end{aligned}$$Above, the loss function ℓ measures the prediction error (MSE in our experiments), $\mathcal {R}$ is a regularization term, and the hyperparameter α trades off the importance of the two. With this notation in place, our proposal aims to minimize the following cost function:$$\begin{aligned} \hat{f}(x) = \underset{f\in \mathcal {F}}{\text {argmin}} \ (1-\lambda) \mathcal {J}_{\text {base}}({\textbf {Y}},{\textbf {X}}; f) + \frac{\lambda }{d} \sum _{j=1}^{d} {\mathcal {D}(z, \tilde{z}_j)}, \end{aligned}$$where $z=\frac{1}{m} \sum _{i=1}^m {T(Y^i,X_j^i,X_{-j}^i;f)}$ and $\tilde{z}_j=\frac{1}{m} \sum _{i=1}^m {T(Y^i,\tilde{X}_j^i,X_{-j}^i;f)}$ are the empirical risks — MSE in our context since $T(\cdot ,\cdot)$ is the squared error (2)—of the model evaluated on the original and synthetic triplets, respectively. The function $\mathcal {D}(z, \tilde{z}_j)\in \mathbb {R}$ measures the difference between z and $\tilde{z}_j$, such that $\mathcal {D}(z, \tilde{z}_j)$ gets smaller as $\tilde{z}_j$ becomes larger than z. Therefore, (5) seeks a model $\hat{f}$ that aligns with the base objective $\mathcal {J}_{\text {base}}$ while creating a better separation between z and $\tilde{z}_j$. The parameter $\lambda \in [0,1)$ sets the relative weight between the two functionals; we present a simple, automatic approach for tuning this parameter in Supplementary Section D. In practice, we define $\mathcal {D}$ as follows: $\mathcal {D}(z,\tilde{z}_j)=\sigma (z-\tilde{z}_j)$, where $\sigma (\cdot ,\cdot)$ is the Sigmoid function, i.e., $\sigma (s)=1/(1+e^{-s})$. Of course, other forms of this function, such as the ratio $(z / \tilde{z}_j)$, can be considered.For ease of reference, Algorithm 1 presents a general approach for minimizing (5). In short, we use stochastic optimization and back-propagation, where each gradient step updates the model parameters by resampling $\tilde{X}_j$ for a subset of features from $\{1,\dots ,d\}$. We sample new dummies at each iteration to reduce algorithmic randomness, which, in turn, increases the stability of the procedure. In Supplementary Section B we provide a weak form of convergence of this algorithm, where the analysis requires $\mathcal {D}(\cdot ,\cdot)$ to be bounded from below. A requirement that is satisfied by our choice to use the Sigmoid function. Algorithm 1 is used in our experiments, implemented with a neural network as the base predictive model. Remark 1 Algorithm 1 implements the RD penalty for all features simultaneously, thereby avoiding the need to fit a different model for each feature separately. This algorithm can be modified to fit a model with an RD penalty defined for any subset of features, and in particular to a specific feature X_j, as we demonstrate in Supplementary Section G.2. 3.3 A scheme for sparse linear regressionNext, we offer an optimization procedure tailored for the choice of sparse linear regression as the base objective, since it is widely used in the feature selection literature and often results in a powerful test (Barber & Candès, 2015; Tansey et al., 2021). To this end, consider the combination of the elastic net objective (Zou & Hastie, 2005) with our MRD loss:$$\begin{aligned} \begin{array}{ll} \hat{\beta } = \underset{\beta }{\text {argmin}} \ (1-\lambda)\bigg (\overbrace{\frac{1}{2m} || {\textbf {X}}\beta - {\textbf {Y}}||^2_2 + \alpha _1 ||\beta ||_1 + \frac{\alpha _2}{2}||\beta ||_2^2}^{\mathcal {J}_{\text {base}}({\textbf {Y}},{\textbf {X}}; f)}\bigg) + \frac{\lambda }{d} \sum _{j=1}^{d} {\mathcal {D}(z(\beta), \tilde{z}_j(\beta))}, \end{array} \end{aligned}$$where $\beta \in \mathbb {R}^d$ is the regression coefficient vector, and $\alpha _1,\alpha _2$ are the elastic net penalty parameters. Above, we use the notations $z(\beta)$ and $\tilde{z}_j(\beta)$ to stress that z and $\tilde{z}_j$ are functions of β. Using variable splitting, we can separate the sparsity promoting penalty from the remaining terms, and solve the resulting optimization problem using the alternating direction method of multipliers ADMM (Boyd et al., 2011); see Supplementary Section C for more details. Following Supplementary Algorithm 2, the advantage of the above splitting strategy is that the minimization with respect to β has a closed-form solution, being a simple projection onto the elastic net ball (Parikh & Boyd, 2014; Yu, 2013). In practice, we run this iterative algorithm until the stopping criteria suggested by Boyd et al. (2011, Section 3.3) is met.We conclude this section by showing that under the assumptions of Proposition 2, the estimated coefficients of the null features $\hat{\beta }_j, j \in \mathcal {H}_0$ obtained by minimizing the MRD objective will be equal to zero. The proof is provided in Supplementary Section A. Proposition 2 Suppose that the population model is linear $Y=X^T\beta +\epsilon$, where $\mathbb {E}[X_jX_k]=0$ for $1 \le k e j \le d$, and $\mathbb {E}[\epsilon X_j] = 0$ for all $1\le j \le d$. Denote by $\hat{\beta }$ a solution to the infinite-data version of the MRD:$$\begin{aligned}\hat{\beta } = \,& \underset{\beta }{\text {argmin}} \ (1-\lambda)\mathbb {E}{[(X^T\beta - Y)^2]}\, + \\ & \quad \frac{\lambda }{d} \sum _{j=1}^d {\sigma (\mathbb {E}[T(Y,\tilde{X}_j,X_{-j}; \beta) - T(Y,X_j,X_{-j}; \beta)]){. \end{aligned}$$Under the above assumptions, for any $0 \le \lambda < 1$, and for all $j \in \mathcal {H}_0$ with $\beta _j=0$, the solution $\hat{\beta }$ satisfies that $\hat{\beta }_j = 0$ for all $j \in \mathcal {H}_0$. Our proposed method inherits the model X assumption that the marginal distribution P_X—required to sample $\tilde{X}_j$—is known: this assumption is needed to guarantee that (1) is a valid p-value (Candès et al., 2018). However, in most real-world applications P_X is unknown and thus must be estimated from the data. Here, dummy features of poor quality may break the validity of model-X randomization tests (Romano et al., 2019; Tansey et al., 2021; Sudarshan et al., 2021), including our proposal. To alleviate this concern, model X tests are equipped with various methodologies to estimate P_X, including a second-moment multivariate Gaussian estimator (Candès et al., 2018), mixture density networks (Tansey et al., 2021), generative adversarial networks (Bellot & van der Schaar, 2019), moment matching networks (Romano et al., 2019), and more (Sesia et al., 2019; Sudarshan et al., 2020; Gimenez et al., 2019; Sudarshan et al., 2021). Furthermore, model-X tests are also equipped with diagnostic tools to improve the confidence in the results obtained by this method when the conditionals are unknown. These tools include goodness-of-fit tests, designed to quantify the quality of the generated dummies, as well as controlled semi-synthetic experiments with simulated response variables that allow the user to compare the obtained empirical FDR with the desired level (Romano et al, 2019, Section 5).Importantly, empirical evidence show that the model-X tests are robust in the sense that the FDR is often controlled in practice, especially when the testing procedure is applied with well-approximated dummy features. This is also corroborated by our experiments on real and simulated data: we practically obtain FDR control even when using the simplest Gaussian approximation to the unknown P_X, highlighting the robustness to model misspecification of the HRT in general, and our proposal, in particular.4.2 Relevant prior artOur proposal is connected to the two-stage "learn-then-test" approach, initiated by the influential work of Gretton et al. (2012) in the context of kernel-based tests. The original idea here is to optimize the kernel parameters in order to maximize the power of the underlying test, where ample evidence shows that this approach is extremely fruitful (Gretton et al., 2012; Liu et al., 2016; Chwialkowski et al., 2016). These contributions are in line with our proposal, suggest learning techniques to improve power, however ours is very different as it is tailored to work in combination with model-X tests.Recently, new deep learning methods for feature selection have been proposed to improve data analysis with non-linear interactions. In particular, Borisov et al. (2019), Lemhadri et al. (2021), Yamada et al. (2020) offer sparsity-promoting layers that can be combined with arbitrary neural network architectures. These techniques can be combined with our MRD framework, as demonstrated in Sect. 5, where we utilize the CancelOut regularizer developed by Borisov et al. (2019). A different but related line of work is that of instance-wise interpretability methods, aiming at identifying a subset of relevant features for a single data point (Chen et al., 2018; Yoon et al., 2019). While this line of work is outside the scope of this paper, we believe these methods may benefit from our MRD approach. For instance, by combining the MRD with the test proposed by Burns et al. (2020), which is capable of finding instance-wise features that a given predictive model considers important.In this section, we study the performance of the proposed MRD approach in a wide variety of settings. These include experiments with linear and nonlinear simulated data of varying signal

case generalization. arXiv preprint arXiv:1911.08731.Sanjabi, M., Ba, J., & Razaviyayn, M., et al. (2018). Solving approximate wasserstein gans to stationarity. arXiv preprint arXiv:1802.08249.Sesia, M., Bates, S., & Candès, E., et al. (2020a). Controlling the false discovery rate in GWAS with population structure. bioRxiv.Sesia, M., Katsevich, E., Bates, S., et al. (2020). Multi-resolution localization of causal variants across the genome. Nature Communications, 11(1), 1–10. Google Scholar Sesia, M., Sabatti, C., & Candès, E. J. (2019). Gene hunting with hidden Markov model knockoffs. Biometrika, 106(1), 1–18.Article MathSciNet MATH Google Scholar Shah, R. D., & Peters, J. (2020). The hardness of conditional independence testing and the generalised covariance measure. The Annals of Statistics, 48(3), 1514–1538.Article MathSciNet MATH Google Scholar Srivastava, N., Hinton, G., Krizhevsky, A., et al. (2014). Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15(1), 1929–1958.MathSciNet MATH Google Scholar Sudarshan, M., Puli, A., & Subramanian, L., et al. (2021). Contra: Contrarian statistics for controlled variable selection. In International conference on artificial intelligence and statistics, PMLR (pp. 1900–1908).Sudarshan, M., Tansey, W., & Ranganath, R. (2020). Deep direct likelihood knockoffs. Advances in Neural Information Processing Systems 33.Tansey, W., Veitch, V., & Zhang, H., et al. (2021). The holdout randomization test for feature selection in black box models. Journal of Computational and Graphical Statistics 1–37.Yamada, Y., Lindenbaum, O., & Negahban, S., et al. (2020). Feature selection using stochastic gates. In International conference on machine learning, PMLR (pp. 10,648–10,659).Yoon, J., Jordon, J., & van der Schaar, M. (2019). INVASE: Instance-wise variable selection using neural networks. In International conference on learning representations.Yu, Y. (2013). On decomposing the proximal map. In Advances in Neural Information Processing Systems 26.Zhang, K., Peters, J., & Janzing, D., et al. (2012). Kernel-based conditional independence test and application in causal discovery. arXiv preprint arXiv:1202.3775.Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. Journal of the Royal Statistical Society, Series B, 67, 301–320.Article MathSciNet MATH Google Scholar Download referencesWe sincerely thank Matteo Sesia for insightful comments about an earlier manuscript draft. S.S and Y.R. were supported by the Israel Science Foundation (Grant 729/21). Y.R. also thanks the Career Advancement Fellowship, Technion, for providing research support.S.S and Y.R. were supported by the Israel Science Foundation (Grant 729/21). Y.R. was also supported by the Career Advancement Fellowship, Technion. The authors declare that they have no conflict of interests. For completeness, we prove Proposition 1 by following the steps from Romano et al, 2020, Proposition 1. We use discrete random variables for simplicity. Denote by \(x' = x_{-j}\) and observe that$$\begin{aligned} \mathbb {P}(Y = y, X_j = x_j, X_{-j} = x')&= \mathbb {P}(Y = y, X_j = x_j \mid X_{-j}= x') \cdot \mathbb {P}(X_{-j} = x') \\ \text {(Since }Y \perp \!\!\! \perp X_j \mid X_{-j}\text {)}\quad&= \mathbb {P}(Y = y \mid X_{-j} = x') \cdot \mathbb {P}(X_j = x_j \mid X_{-j} = x')\\&\qquad \cdot \mathbb {P}(X_{-j} = x') \\ \text {(By construction of }\tilde{X}_j\text {)}\quad&= \mathbb {P}(Y = y \mid X_{-j} = x') \cdot \mathbb {P}(\tilde{X}_j = x_j \mid X_{-j} = x') \\&\qquad \cdot \mathbb {P}(X_{-j} = x') \\ \text {(Since }Y \perp \!\!\! \perp \tilde{X}_j \mid X_{-j}\text {)}\quad&= \mathbb {P}(Y = y, \tilde{X}_j = x_j \mid X_{-j} = x') \cdot \mathbb {P}(X_{-j} = x') \\&= \mathbb {P}(Y = y, \tilde{X}_j = x_j, X_{-j} = x'). \end{aligned}$$\(\square\) Proof of Proposition 2 Following Corollary 1, for any null feature \(\beta _j = 0\), we know that the following holds$$\begin{aligned}&\mathbb {E}[T(Y,\tilde{X}_j,X_{-j}; \hat{\beta }) - T(Y,X_j,X_{-j}; \hat{\beta })] = 0, \end{aligned}$$for any value of \(\hat{\beta }_j\). We now turn to study the effect of \(\hat{\beta }_j, j\in \mathcal {H}_0\) on a non-null feature \(k \in \mathcal {H}_1\), by re-writing the test statistic:$$\begin{aligned} \mathbb {E}[T(Y,{X}_k,X_{-k}; \hat{\beta })]&= \mathbb {E}[(Y - {X}_j \hat{\beta }_j - {X}_k \hat{\beta }_k - {X}_{-(j,k)}^T\hat{\beta }_{-(j,k)})^2] onumber \\&= \mathbb {E}[({X}_j {\beta }_j + {X}_k {\beta }_k + {X}_{-(j,k)}^T{\beta }_{-(j,k)} onumber \\&\quad +\epsilon - {X}_j \hat{\beta }_j - {X}_k \hat{\beta }_k - {X}_{-(j,k)}^T\hat{\beta }_{-(j,k)})^2] onumber \\&= \mathbb {E}\left[({X}_k ({\beta }_k - \hat{\beta }_k) + {X}_{-(j,k)}^T({\beta }_{-(j,k)}\right. onumber \\&\quad \left. - \hat{\beta }_{-(j,k)}) + \epsilon - {X}_j \hat{\beta }_j)^2\right] onumber \\&= \mathbb {E}[({X}_k ({\beta }_k - \hat{\beta }_k))^2] + \mathbb {E}[({X}_{-(j,k)}^T({\beta }_{-(j,k)} - \hat{\beta }_{-(j,k)}))^2] onumber \\&\quad +\mathbb {E}[\epsilon ^2] + \hat{\beta }_j^2\mathbb {E}[{X}_j^2]. \end{aligned}$$ (A1) The third equality uses the fact that \(\beta _j=0\) and the last one relies on the assumptions that \(\mathbb {E}[X_j,X_k] = 0\) for all \(1 \le j e k \le d\) and \(\mathbb {E}[X_j,\epsilon] = 0\) for all \(1 \le j \le d\). Similarly, since \(\mathbb {E}[{X}_j \tilde{X}_k] = 0\) we have also that$$\begin{aligned} \mathbb {E}[T(Y,\tilde{X}_k,X_{-k}; \hat{\beta })]&= \mathbb {E}[({X}_k{\beta }_k - \tilde{X}_k\hat{\beta }_k))^2] + \mathbb {E}[({X}_{-(j,k)}^T({\beta }_{-(j,k)} - \hat{\beta }_{-(j,k)}))^2] + \\ {}&\quad \ \mathbb {E}[\epsilon ^2] + \hat{\beta }_j^2\mathbb {E}[{X}_j^2]. \end{aligned}$$As a result, we get that \(\hat{\beta }_j, j\in \mathcal {H}_0\) does not influence the risk discrepancy of the null and non null features. To complete the proof, we now show that any choice of \(\hat{\beta }_j e 0\) can only increase the MSE term in the objective function. Applying similar steps as in (A1), we get$$\begin{aligned} \mathbb {E}[(Y - X^T\hat{\beta })^2] = \mathbb {E}[({X}_{-j}^T({\beta }_{-j} - \hat{\beta }_{-j}))^2] + \mathbb {E}[\epsilon ^2] + \hat{\beta }_j^2\mathbb {E}[{X}_j^2], \end{aligned}$$which is minimized for the choice of \(\hat{\beta }_j = 0\). \(\square\) Appendix B Supplementary analysis of the optimization algorithmIn this section, we follow Romano et al. (2019), Sanjabi et al. (2018) and provide a weak form of convergence to the general learning scheme presented in Algorithm 1 of the main manuscript. Denote by \(f_{\theta _k}\) a predictive model, parameterized by \(\theta _k\) (e.g., the weights and biases of a neural network), at step k of the optimization procedure. Recall that in Algorithm 1, in each iteration we choose a random subset of indices \(\mathcal {P}_k\) and generate a fresh dummy copy \(\tilde{X}_{j}^i\) of \({X}_{j}^i\) for all \(j \in \mathcal {P}_k\). For ease of notation, we denote by \(\tilde{{\textbf {X}}}_k \in \mathbb {R}^{m \times |\mathcal {P}|}\) the dummy matrix with the new \(\tilde{X}_{j}^i\) as its (i, j)’s entry. Thus, conditional on \((\mathcal {P}_k,\tilde{{\textbf {X}}}_k)\) the objective function$$\begin{aligned} \mathcal {J}({\textbf {X}},{\textbf {Y}},f_{\theta _k},\mathcal {P}_k,\tilde{{\textbf {X}}}_k) =(1-\lambda) \mathcal {J}_{\text {base}}({\textbf {Y}},{\textbf {X}}; f_{\theta _k}) + \frac{\lambda }{|\mathcal {P}_k|} \sum _{j \in \mathcal {P}_k} {\mathcal {D}(z(f_{\theta _k}), \tilde{z}_j(f_{\theta _k}))}, \end{aligned}$$is a deterministic function of \(\theta _k\). Now, we can define$$\begin{aligned} \mathcal {J}_{\theta _k}=\mathbb {E}_{\mathcal {P}_k,\tilde{{\textbf {X}}}_k}[{\mathcal {J}({\textbf {X}},{\textbf {Y}},f_{\theta _k},\mathcal {P}_k,\tilde{{\textbf {X}}}_k) \mid \theta _k}] \end{aligned}$$ (B2) and denote by \(abla \mathcal {J}_{\theta _k}\) its gradient with respect to \(\theta _k\). As depicted in Algorithm 1, we estimate \(abla \mathcal {J}_{\theta _k}\) by sampling one realization of \(\mathcal {P}_k\) and \(\tilde{{\textbf {X}}}_k\):$$\begin{aligned} g_t = abla \mathcal {J}({\textbf {X}},{\textbf {Y}},f_{\theta _k},\mathcal {P}_k,\tilde{{\textbf {X}}}_k), \end{aligned}$$ (B3) resulting in an unbiased estimate of \(abla \mathcal {J}_{\theta _k}\). Since all random variables are observed at the kth step, \(g_t\) from (B3) is deterministic and backpropogation can be applied to update \(\theta _{k+1} \leftarrow \theta _{k} - \eta g_t\), where \(\eta\) is the learning rate (step size). Our analysis assumes that there exists a finite Lipschitz constant L, which satisfies the following relation for all \(\theta ',\theta ''\):$$\begin{aligned} ||abla \mathbb {E}_{\mathcal {P},\tilde{{\textbf {X}}}}[{\mathcal {J}({\textbf {X}},{\textbf {Y}},f_{\theta },\mathcal {P},\tilde{{\textbf {X}}})} \mid \theta '}] -abla \mathbb {E}_{\mathcal {P},\tilde{{\textbf {X}}}}[{\mathcal {J}({\textbf {X}},{\textbf {Y}},f_{\theta },\mathcal {P},\tilde{{\textbf {X}}})} \mid \theta ''}]||_2 < L||\theta '-\theta ''||_2. \end{aligned}$$Denote \(\Delta = \frac{2}{L} \sup (\mathcal {J}_{\theta _1} -\mathcal {J}^*)\), where \(\mathcal {J}_{\theta _1}\) is the objective obtained by the initialized model, i.e., by assigning \(\theta _1\) in (B2), and \(\mathcal {J}^*\) is a lower bound uniformly the value of \(\mathcal {J}_{\theta _k}\), for all k. The supremum is taken over all possible values of \(\mathcal {P},\tilde{{\textbf {X}}}\) and \(\theta _1\). Suppose we can uniformly bound from below the RD term, which holds for our choice of \(\mathcal {D}\) from Sect. 3.2, since \(\mathcal {D}(z,\tilde{z}_j) = \sigma (z-\tilde{z}_j) > 0\). Suppose further that we can uniformly lower bound the objective of the base predictive model; for example, the MSE loss is bounded from below by 0. With these notations in place, we can immediately invoke Theorem 2 from Romano et al. (2019), which is stated below formally for convenience. In short, Theorem 3 provides us a weak form of convergence as it shows that the average squared norm of the gradient \(\mathbb {E}[||abla \mathcal {J}_{\theta _k} ||_2^2 \mid \theta _1]\) decreases as \(\mathcal {O}(1/\sqrt{K})\); a detailed discussion on this result is provided by Romano et al. (2019); see also Sanjabi et al. (2018). Theorem 3 (Theorem 2, Romano et al. 2019) Consider a fixed training set \({\textbf {X}},{\textbf {Y}}\), and use the notation from above. Assume$$\begin{aligned} \mathbb {E}[||g_t - abla \mathcal {J}_{\theta _k}||_2^2 \mid \theta _k] \le \sigma ^2, \quad \forall k \le K, \end{aligned}$$for some \(\sigma \in \mathbb {R}\). Then for any initial \(\theta _1\) and a suitable value of \(\Delta\),$$\begin{aligned} \frac{1}{K} \sum _{k=1}^K {\mathbb {E}[||abla \mathcal {J}_{\theta _k} ||_2^2 \mid \theta _1]} \le \frac{1}{K} \frac{L\Delta }{\eta (2-L\eta)}+\frac{L\sigma ^2\eta }{(2-L\eta)}. \end{aligned}$$Specifically, choosing \(\eta =\min \{\frac{1}{L},\frac{\Delta _0}{\sigma \sqrt{K}}\}\) for some \(\eta _0 > 0\) gives$$\begin{aligned} \frac{1}{K} \sum _{k=1}^K {\mathbb {E}[||abla \mathcal {J}_{\theta _k} ||_2^2 \mid \theta _1]} \le \frac{L^2\Delta }{K} + \bigg (\eta _0 +\frac{\Delta }{\eta _0} \bigg) \frac{L\sigma }{\sqrt{K}}. \end{aligned}$$ Appendix C Supplementary details on fitting sparse linear regression modelsAlgorithm 2 presents our specialized learning scheme for linear regression model. This algorithm minimizes the following objective:$$\begin{aligned} \hat{\beta } =&\underset{\beta , \text {v}}{\text {argmin}} \ (1-\lambda)\bigg (\frac{1}{2m} || {\textbf {X}}\textrm{v} - {\textbf {Y}}||^2_2 + \alpha _1 ||\beta ||_1 + \frac{\alpha _2}{2}||\beta ||_2^2\bigg) + \frac{\lambda }{d} \sum _{j=1}^{d} {\mathcal {D}(z(\textrm{v}), \tilde{z}_j(\textrm{v}))} \ {}&\text {subject to} \quad \beta =\textrm{v}, \end{aligned}$$using ADMM (Boyd et al., 2011).Section 5 of the main manuscript compares the performance of several regression models on synthetic and real data sets. Below, we provide additional details on each method. Lasso: we fit a lasso regression model using Python’s sklearn package and tuned the lasso penalty parameter via 5-fold cross-validation. Elastic net: an elastic net model is fitted using the same Python package and training strategy described above. Neural network (NNet): the network architecture consists of two fully connected layers with a hidden dimension of 16; the model is fitted using Adam optimizer (Kingma & Ba, 2014) with a fixed learning rate of 0.005. To avoid over-fitting, a dropout regularization (Srivastava et al., 2014) is deployed, with a rate of 0.5. We also use CancelOut (Borisov et al., 2019) regularization to promote sparsity in the entry layer.3 In Sect. 5.1, we train the network for 60 epochs, whereas in Sect. 5.2 we reduced this number due to over-fitting. Specifically, we use 15 epochs for \(c\in \{0.5,0.625,0.75\}\), and 10 epochs for \(c\in \{1,1.25\}\), where c is the signal strength of the semi-synthetic data. MRD lasso: we fit the model by minimizing (5) in combination with lasso regularization using Algorithm 2. The lasso penalty parameter is equal to the one used in lasso. The MRD penalty \(\lambda\) is set to be equal to \(\min \{0.8, 0.8\cdot \texttt {MSE-Validation}\}\), where MSE-Validation is the average MSE evaluated on the 5 folds used to tune the hyperparameter of the baseline lasso model. Regarding the hyperparameter of the ADMM framework, we set \(\rho =1\) and the stopping criteria parameters \(\epsilon ^{\textrm{rel}} = 10^{-3},\epsilon ^{\textrm{abs}}=5\cdot 10^{-4}\); we deployed the stopping rule suggested in (Boyd et al, 2011, Section 3.3). MRD elastic net: here, we combine the elastic net penalty with our RD loss and use the same training strategy described in MRD lasso. MRD neural network: we choose the same architecture, optimizer, and hyperparameters outlined in neural network, and fit the model as described in Algorithm 1. We set the MRD penalty \(\lambda\) to be equal to \(\min \{0.8, 0.8\cdot \texttt {MSE-Validation}\}\), where the MSE-Validation is evaluated as follows. First, we split the training data into training (80%) and validation (20%) sets. Then, we fit the base neural network model (without the penalty) on the training data, and use this model to compute the MSE-Validation on the validation set. We run all the numerical experiments on our local cluster, which consists of two AMD servers with 24 CPU cores each.Appendix E Supplementary details on the effect of \(\lambda\) Section D of the Supplementary Material presents a simple way to choose the penalty parameter \(\lambda\). Here, we demonstrate the impact of different values of \(\lambda\) on the performance of the MRD approach. Figure 5 presents the empirical power obtained by using the MRD lasso, as a function of the penalty parameter \(\lambda\). The vertical green line represents the choice of our approach for choosing \(\lambda\). Here, we use the same non-linear synthetic data as in Sect. 5.1.1, with three different signal amplitude values, \(c=0.14\), \(c=0.15\) and \(c=0.18\), representing low, medium and high power regimes. As can be seen, in the low and medium power regimes, the choice of every presented value of \(\lambda\) results in higher power than the base model (i.e., \(\lambda =0\)). However, in the high power regime, higher values of \(\lambda\) reduces the power of the base models. This phenomenon corroborates our approach to choose \(\lambda\) to be proportional to the prediction error of the base model, as presented in Supplementary Section D.Following that figure, we can also see that our proposed automatic approach for choosing \(\lambda\), returns a penalty parameter that is fairly close the best one.Fig. 5The effect of the MRD regularization on power, demonstrated using a synthetic data with correlated multivariate Gaussian features and a simulated response that follows a non-linear model. Each panel displays the empirical power (evaluated by averaging over 50 independent experiments) as a function of the MRD penalty parameter \(\lambda\), using lasso as the base predictive model. The green vertical line represents the value obtained by our proposed automatic approach for choosing \(\lambda\). Left: low power regime for which the signal amplitude \(c=0.14\). Middle: low power regime for which the signal amplitude \(c=0.15\). Right: high power regime for which the signal amplitude \(c=0.18\)In Sect. 3.3 we describe our learning procedure for the choice of sparse linear regression as the base objective. Here, we demonstrate the impact of the MRD approach on the estimated regression coefficients. Figure 6a presents the average of the estimated regression coefficients of lasso and MRD lasso from the experiment presented in Sect. 5.1.1. As can be seen, on average, the estimated regression coefficients of the null features that correspond to the MRD lasso are slightly larger than those of lasso, but both are relatively close to zero. By contrast, the coefficients of the non-null features have a larger magnitude, where it is evident that the MRD approach further increases the coefficients’ magnitude. Figure 6b also presents the magnitude of the estimated coefficients of MRD lasso that correspond to the non-null features as a function of lasso coefficients. Each color is for a different important feature, so there are 100 points per feature, one for a different trial. We can see that the MRD lasso’s coefficients are with larger magnitude than those of lasso, where the amplification of the lower coefficients is more significant than of the higher ones. Specifically, it can be observed that in several cases lasso nullifies coefficients of important features, whereas MRD lasso yields non-zero values (the opposite is not true).Fig. 6Comparison between MRD lasso regression coefficients and vanilla lasso regression coefficients. The predictive models are fitted on the synthetic non linear data from Sect. 5.1.1. Left: the regression coefficients of lasso and MRD lasso for all features. Each point in this panel is obtained by averaging the estimated regression coefficient over 100 independent trials. Green and orange dots represent null features; blue and red dots represent non-null features. Right: scatter plot of the magnitude of MRD lasso regression coefficients as a function of the vanilla lasso regression coefficients for non-null features. Each color represent a different non-null feature (Color figure online)Linear \(Y\mid X\) and additional comparisonsIn Table 1 of the main manuscript we compared the performance of MRD lasso, MRD elastic net and MRD NNet to their base models by varying the signal strength while fixing the target FDR level to \(q=0.2\). For completeness, we provide here the root mean squared error (RMSE) and the empirical FDR obtained by each model as well as additional comparisons to other baseline methods. We also present here the empirical power, FDR and RMSE of our experiments in the linear setting, where \(Y \mid X\) follows M1. Table 4 focuses on the linear setting, where we include a ridge regression model in addition to lasso and elastic net. The penalty parameter of the ridge regression model is tuned via 5-fold cross-validation. Following that table, we can see that the empirical power obtained by the ridge model does not exceed the ones achieved by our proposed MRD methods. Observe also that the RMSE scores obtained by MRD lasso and MRD elastic net are similar to that of the corresponding baseline models. Turning to the non-linear case, Table 5 summarizes the performance of random forest and kernel ridge regression in addition to lasso, elastic net, and neural network. We fit the kernel ridge regression model using the sklearn software package, where we choose a polynomial kernel of degree 3 and a fixed penalty parameter that equal to 1. The random forest model is also fitted using sklearn, where we choose the software’s default hyperparameters except the number of trees and the maximum depth of each tree; we set both to 100. In terms of power, the kernel ridge model outperforms the baseline models, however, it does not exceed the performance of MRD lasso and MRD elastic net. The random forest model is less competitive and yields lower power compared to the other methods, across all experiments. Regarding the RMSE, overall we can see that the MRD models perform similarly to their baseline counterparts, except the case of a neural network base model. There, the RMSE of the MRD neural network is slightly improved.Table 4 Synthetic linear data with varying signal strength cTable 5 Complementary results to those presented in Table 1. All standard errors are below 0.02Fig. 7Synthetic experiments with correlated multivariate Gaussian features and a simulated response that follows a linear model with a constant signal strength \(c=0.12\). Each graph displays the empirical FDR and power (evaluated by averaging over 100 independent experiments) as a function of the target FDR level q. Left: MRD lasso and lasso. Right: MRD elastic net and elastic netFig. 8Synthetic experiments with correlated multivariate Gaussian features and a simulated response that follows a non-linear model. The signal strength is fixed and equals to \(c=0.14\). Left: MRD lasso and lasso. Middle: MRD elastic net and elastic net. Right: MRD neural network (NNet) and NNet. The other details are as in Fig. 7Fig. 9Testing for a single hypothesis on synthetic data with correlated multivariate Gaussian features and a simulated response that follows a linear model. The significance level is fixed and equals to \(\alpha =0.05\). The empirical power is presented as a function of the signal strength c, and evaluated by averaging over 100 independent experiments. Left: MRD lasso and lasso. Right: MRD elastic net and elastic netFig. 10Testing for a single hypothesis on synthetic data with correlated multivariate Gaussian features and a simulated response that follows a non-linear model. Left: MRD lasso and lasso. Middle: MRD elastic net and elastic net. Right: MRD neural network (NNet) and NNet. The other details are as in Fig. 9Fig. 11Optimizing and testing for a single hypothesis on synthetic data with correlated multivariate Gaussian features and a simulated response that follows a non-linear model. The significance level is fixed and equals to \(\alpha =0.1\). The empirical power is presented as a function of the signal strength c, and evaluated by averaging over 100 independent experiments. Left: MRD lasso and lasso. Right: MRD elastic net and elastic netPower as a function of the target FDRThe experiments presented above focus on feature selection with a fixed target FDR level. Figures 7, 8 present additional comparisons in which we vary the target FDR level while keeping the signal amplitude fixed. Following Fig. 7, which corresponds to the linear case (i.e., \(Y \mid X\) follows M1), we can see that the MRD models outperform their baseline counterparts in terms of power. Observe that the FDR is controlled in all cases, where the base models are more conservative than their MRD versions. A similar trend is illustrated in Fig. 8 that represents the nonlinear setting (where \(Y \mid X\) follows M2). Here, we include a neural network model in addition to the sparse linear methods. Notice that the MRD neural network model attains better power than the baseline method, while the empirical FDR of the two is similar, both fall below the target level.p-value analysisHerein, we study the behaviour of the p-values produced by the MRD approach. Figure 12 presents Q-Q plots of the p-values from the experiments summarized in Table 1, focusing on the case where \(c=0.14\) with lasso as the baseline model. The left panel of Fig. 12 presents the empirical quantiles of the p-values that correspond to the null features as a function of the quantiles of the uniform distribution on the [0, 1] segment. As can bee seen, the HRT p-values for the null features are valid but conservative both for lasso and MRD lasso where the distribution of the latter p-values are closer to the uniform distribution. The right panel of Fig. 12 presents an analogous plot but for the non-null features. Here, when combining HRT with MRD lasso we get p-values with lower quantiles that the ones of lasso, and both lie below the quantiles of the uniform distribution. Importantly, the MRD lasso leads to smaller p-values than lasso for the non-null features, demonstrating the advantage of the MRD approach.Fig. 12Q-Q plots of the p-values produced in the experiment from Table 6 compared to the uniform distribution U[0, 1]. Left: p-values correspond to the null features. Right: p-values correspond to the non-null featuresWe now analyze the effect of the MRD approach when testing for a single non-null hypothesis. Specifically, we choose at random a non-null j from \(\mathcal {H}_1\), and test for \(H_{0,j}\) at level \(\alpha =0.05\). We first consider the p-value evaluated via the HRT, using the same models and data from Tables 4, 5. Figure 9 focuses on the case for which \(Y \mid X\) follows a linear model (i.e., M1), presenting the power of lasso, elastic net, and their MRD versions as a function of the signal strength c. As can be seen, the results are aligned with the FDR experiments in the sense that the MRD approach increases the power of the test. Figure 10 repeats the same experiment in the non-linear case (where \(Y \mid X\) follows M2). Here, once more, an improvement in power is achieved by the MRD approach. Notice that the gain in performance is not consistent for the base neural network model.Next, we demonstrate the advantage of the MRD when optimizing for a specific feature j. To this end, we fit a model via Algorithm 1 but set \(\mathcal {P} = \{j\}\) in line 2 of Algorithm 1. We use the same synthetic experimental setting with a non linear data model (M2) as in the experiments from Sect. 5.1.2. Specifically, we fit MRD lasso and MRD elastic net for a single arbitrary non-null feature instead of all the features simultaneously. Figure 11 presents the power of lasso, MRD lasso, elastic net, and MRD elastic net as a function of the signal strength c. As can be seen, the power of the MRD models is higher than that of the baseline models.1.3 G.3 Additional experiment with interaction modelIn this experiment, we set the auto-correlation parameter \(\rho =0.25\) and choose the Y|X to be \(Y = \sum _{j=1}^{15} { X_{2j}X_{2j-1} } +\epsilon\) (i.e., \(Y \mid X\) follows M2), such that in total there are 30 non-null elements out of total \(d=100\) features. Figure 2 presents the empirical power and FDR (target level of \(q = 0.2\)) obtained by a base model formulated as a neural network. We use the same network architecture and training procedure as described in Section D, but with total \(n=4000\) observations (\(m=2000\)), learning-rate of 0.006, fit the model to a total of 200 epochs, and set the CancelOut regularization parameter to 0.02. (In this experiment, linear predictive models attain zero power, and hence omitted.)1.4 G.4 Experiments with sum of sines modelIn this section, we provide additional experiment with a different conditional modeling \(Y \mid X\). Here, we use the same synthetic experiment setting as in the experiments with varying signal strength from Sect. 5.1.2, however generate \(Y \mid X\) that follows M3. Table 6 presents the empirical power and FDR (q = 0.2) obtained when using the elastic-net and lasso as the base predictive models. Following that table, the MRD achieves a gain in power compared to the base model, and the FDR is controlled.Table 6 Synthetic experiments with correlated multivariate Gaussian features and a simulated response that follows a non-linear model with \(g(X) = \sum _{j \in \mathcal {H}_1} {\sin {(X_j\beta _j)}}\)A limitation of the model-X randomization test is that \(P_X\) must be known in order to sample the dummy features \(\tilde{X}_j\), which are also used to formulate our RD loss. In this section, we demonstrate that the feature selection pipeline applied with the proposed MRD method is fairly robust to unknown or estimated distributions. For ease of computations, we conduct our experiments by choosing lasso or elastic net as the base models, fitted in the same way described in Section D.Varying the estimation quality of the conditionalsWe begin with an experiment that demonstrates the effect of the density estimation on the empirical FDR. We generate features as in Sect. 5.1, but with an auto-correlation coefficient \(\rho =0.5\), higher than the one used in the synthetic experiments from that section. Here, the model of \(Y \mid X\) follows M2 with a fixed signal strength \(c=0.16\), where we use lasso as a base predictive model. To control the quality of the estimation of \(P_X\), we sampled the dummy features \(\tilde{X}_j\) from \(Q_{X_j \mid X_{-j}}\), where \(Q_X\) also follows a multivariate Gaussian however with an auto-correlation coefficient \(\hat{\rho }\) that can differ from the true \(\rho =0.5\). We evaluate the quality of the dummies \({\tilde{X}}_j \sim Q_{X_j \mid X_{-j}}\) using the covariance goodness-of-fit diagnostic, presented by Romano et al, 2019, Section 5. In short, this diagnostic presents an unbiased estimate of the distance between \(\textrm{Cov}(X_j,X_{-j})\) and \(\textrm{Cov}(\tilde{X}_j,X_{-j})\), where larger values imply worse estimation of the conditional. The left panel of Fig. 13 depicts the sum of the covariance diagnostics over \(j=1,\dots ,p\). Each point in that graph is the average value taken over 50 independent data sets, where each is of size 10, 000 samples. As can be seen, the value of the sum covariance diagnostics is close to zero when \(\hat{\rho } = 0.5\), as expected, and gets higher as \(\hat{\rho }\) is far from the true value of \(\rho =0.5\).Next, we turn to study the effect of sampling inaccurate dummies on the performance of our proposed method. To this end, we use the inaccurate dummies (sampled from \(Q_{X_j \mid X_{-j}}\)) both for fitting the MRD model and for computing the test p-values. The right panel of Fig. 13 depicts the empirical power and FDR of both MRD lasso and its base version, evaluated on 50 independent data sets of size \(n=1000\) each. Following that figure, we observe that the MRD lasso is slightly more sensitive to the use of inaccurate dummy features, however a violation in the FDR control occurs only when \(\hat{\rho } \ge 0.8\) for which the sum of the covariance diagnostics is above 10. This experiment illustrates that our method is fairly robust to inaccurate estimation of \(P_X\), where invalid results may be obtained when this estimation is of poor quality. Importantly, this statement holds true for all model-X randomization tests, as the generated p-values are valid only when \(P_X\) is known. In terms of power, the MRD lasso presents, once again, a solid gain in performance compared to lasso when \(\hat{\rho }=\rho\), where the FDR is controlled. (Recall that the correlation structure here is more significant than the synthetic experiments from Sect. 5.1, in which \(\rho \in \{0.1,0.25\}\).)Fig. 13Experiment with dummy features whose quality varies with \(\hat{\rho }\). Here, the true \(P_X\) follows a multivariate Gaussian distribution with with auto-correlation \(\rho =0.5\). Left: sum of covariance diagnostics over all \(j=1,\dots ,p\) (lower values correspond to more accurate dummies) as a function of \(\hat{\rho }\). Right: empirical power and FDRMultivariate GaussianConsider the experimental setup from the experiments with varying signal strength in Sect. 5.1, where \(X \sim \mathcal {N}(\mu ,\Sigma)\) is drawn from an auto-regression model. There, we fit the MRD models and apply the HRT by drawing \(\tilde{X}_j\) from the true \(P_{X_j \mid X_{-j}}\). Here, we conduct similar experiments, but sample \(\tilde{X}_j\) from \(\hat{P}_{X_j \mid X_{-j}}\), where \(\hat{P}_X \sim \mathcal {N}(\hat{\mu },\hat{\Sigma })\) with mean \(\hat{\mu }\) and covariance matrix \(\hat{\Sigma }\) that are estimated from the observed data. Following Figs. 14, 15, we can see that all predictive models—including ours—result in FDR below the nominal 20% level, both in the linear and non-linear settings. Notice that our MRD approach consistently improves the power of each base model, which is in line with the experiments from Sect. 5.1.Fig. 14Robustness experiments with correlated multivariate Gaussian features and a simulated response that follows a linear model with varying signal strength c. The mean and covariance of \(P_X\) are estimated from the data. Left: power (top) and FDR (bottom) of MRD lasso and lasso. Right: power (top) and FDR (bottom) of MRD elastic net and elastic net. The performance metrics are averaged over 100 independent experimentsFig. 15Robustness experiments with correlated multivariate Gaussian features and a simulated response that follows a non-linear model with varying signal strength c. The other details are as in Fig. 14Gaussian mixture modelWe now present a more challenging experiment in which X is drawn independently from a multivariate Gaussian mixture model:$$\begin{aligned} X \sim \left\{ \begin{array}{lll} \mathcal {N}(0,\Sigma _1), &{} \quad \mathrm {w.p} \quad \frac{1}{3}, \\ \mathcal {N}(0,\Sigma _2), &{} \quad \mathrm {w.p} \quad \frac{1}{3}, \\ \mathcal {N}(0,\Sigma _3), &{} \quad \mathrm {w.p} \quad \frac{1}{3}. \end{array}\right. \end{aligned}$$Above, \(\Sigma _1,\Sigma _2\) and \(\Sigma _3\) have the same correlation structure described in Sect. 5.1, with auto-correlation coefficients \(\rho _1 = 0.1\), \(\rho _2 = 0.2\), and \(\rho _3 = 0.3\), respectively. Similar to the experiments presented above, we assume that X follows a multivariate Gaussian and we estimate its parameters from the observed data. This allows us to study the robustness of the proposed MRD approach to model misspecification. We follow the experimental protocol described in the experiments with varying signal strength from Sect. 5.1.2 (with M1 for the non-linear setting), and summarize the results in Figs. 16, 17. As can be seen, the FDR obtained by all models is below the desired 20% level both in the linear and non-linear settings, indicating the robustness of the feature selection procedure. Moreover, our MRD approach yields higher power than the base models, demonstrating that the proposed method is also effective when the distribution of X is unknown.Fig. 16Robustness experiments with correlated multivariate Gaussian mixture features and a simulated response that follows a linear model with varying signal strength c. The other details are as in Fig. 14Fig. 17Robustness experiments with correlated multivariate Gaussian mixture features and a simulated response that follows a non-linear model with varying signal strength c. The other details are as in Fig. 14Multivariate Student’s t-distributionIn the experiments from previous paragraphs, we use a simple Gaussian approximation of \(P_X\), and we always obtain a control of the FDR. Now, we turn to a scenario where such an approximation does not result in FDR control, and demonstrate that using a better density estimation for \(P_X\) addresses tackles this limitation. To this end, we generate \(n=1000\) samples of X, where X follows a multivariate Student’s t-distribution with zero mean, covariance matrix with entries \(\Sigma _{i,j} = \rho ^{|i-j|}\) where \(\rho =0.1\), and \(u = 5\) degrees of freedom; refer to (Romano et al, 2019, Section 6.5) for the precise definition of this data generating function. We begin this experiment by sampling Y that follows a linear model (i.e., M1) of a varying signal strength c, as described in Sect. 5.1 of the main manuscript, where we use the same multivariate Gaussian approximation of \(P_X\) as in the previous paragraphs. The left panel of Fig. 18 shows the empirical power and FDR evaluated on 50 independent data sets, comparing the performance of an elastic net model and its MRD version. Next, we generate the response Y using the non linear model M2, and again vary the signal strength c as described in Sect. 5.1.2 of the main manuscript. As before, we approximate the distribution of X by fitting a multivariate Gaussian on the observed data. The middle panel of Fig. 18 presents the empirical power and FDR of this non linear setting, demonstrating how the MRD elastic net fails to control the FDR while the base elastic net model is below the nominal FDR rate. This violation is due to the inccurate estimation of \(P_X\) provided by the naive multivariate Gaussian fit. To alleviate this, we now turn to approximate \(P_X\) using a Gaussian Mixture Model (GMM) (Reynolds, 2009) with 3 components, where we use Python’s sklearn package to fit such a GMM model. The right panel of Fig. 18 presents the empirical power and FDR obtained by sampling the dummy features from the fitted GMM model (Gimenez et al., 2019), instead of a naive multivariate Gaussian, where we use the same non-linear setting as in the middle panel. Following the right panel of that figure, we can see that the FDR is controlled across all signal strength values, showing that by using better techniques to approximate \(P_X\) we get more reliable results. This is in line with many other techniques proposed in the model-X literature that offer practical solutions to sampling more accurate dummies, as discussed in Sect. 4.1 of the main manuscript. Lastly, in terms of power, the one can see that the MRD models outperform their base versions across the board, emphasizing the superiority of our proposal.Fig. 18Experiments with correlated multivariate Student t-distribution features and a simulated response with varying signal strength c. Power and FDR are evaluated on 50 independent data sets. Left: response that follows a linear model, where the dummies are sampled from a naive multivariate Gaussian approximation of \(P_X\). Middle: response that follows a non linear model, where the dummies are also sampled from a naive multivariate Gaussian. Right: response that follows a non linear model, where the dummies are sampled from a GMM approximation of \(P_X\)In this section, we study the effect of the correlation between features on the proposed MRD approach. To this end, we follow the experimental setting described in Sect. 5.1, where \(Y\mid X\) follows M2 with a fixed signal strength \(c=0.14\) and \(m=400\), and vary the auto-correlation parameter \(\rho\). Figure 19 presents the empirical power and FDR, evaluated over 100 independent data sets, as a function of \(\rho\) for lasso, elastic net and their MRD versions. As can be seen, the power of all methods decreases as \(\rho\) increases, indicating the challenge of testing for conditional independence with highly correlated features. However, the MRD approach results in higher power for almost every \(\rho\) while maintaining the FDR under control, demonstrating the advantage of the MRD approach even when working with correlated features.Fig. 19Experiments with correlated features that follow \(\mathcal {N}(0,\Sigma)\) where \(\Sigma _{i,j}=\rho ^{|i-j|}\) with varying auto-correlation parameter \(\rho\), and a simulated response. Power and FDR are evaluated on 100 independent data setsIn this section, we use simulated data to compare the performance of CV-HRT (Tansey et al, 2021, Algorithm 4) to that of dCRT (Liu et al., 2020) and of the model-X knockoffs (Candès et al., 2018). (Recall that the former relies on cross-validating and thus avoids the naive data-splitting of the vanilla HRT procedure.) To this end, we generate \(X \in \mathbb {R}^{d}\) that follows an autoregression model with correlation coefficient \(\rho =0.6\), and set \(d=100\). Then, we generate a response variable \(Y=(X^T\beta + \epsilon)^3\), where \(\epsilon \sim \mathcal {N}(0,1)\) is a noise component, and \(\beta \in \mathbb {R}^d\) is a sparse vector whose 30 first entries are equal to \(c=1.5\); we set the rest 70 elements in \(\beta\) to zero. We choose the elastic net to be the base predictive model for both CV-HRT and its MRD version, and compare the performance of these two tests to dCRT and to the model-X knockoffs. The CV-HRT is implemented by setting the number of folds to be equal to \(K=20\). We implemented the dCRT in Python, by following the official R package provided by the authors4. we run the dCRT method using the default choice of parameters, provided in the official R package. We run the model-X knockoffs with a second order estimation for generating the knockoffs, by using the package provided by Romano et al. (2019) with its default choice of parameters.5. Table 7 compares the performance of the above four frameworks (averaged over 50 trials) for a varying number of samples n, where the target FDR level is fixed and equals to \(q=0.1\). Following that table, we can see that the gap between CV-HRT and the dCRT is relatively small, and the MRD approach closes this gap even further, achieving essentially the same power as dCRT. The model-X knockoffs achieves lower power than all other frameworks. In all experiments, the empirical FDR is controlled, where the MRD version of CV-HRT tends to have a lower empirical FDR compared to dCRT and the model-X knockoffs. Lastly, we note that CV-HRT requires fitting K different models (20 in this experiment), less than the d leave-one-covariate-out models that are required to implement the dCRT test. Therefore, in general, the computational complexity of CV-HRT may be lower than that of the dCRT.Table 7 Synthetic experiments with correlated multivariate Gaussian features of dimension \(d=100\), and a simulated response that follows a non-linear model with varying number of samples nIn Sect. 5.2 of the main manuscript, we analyze which genetic variations are linked to changes in human immunodeficiency virus drug-resistance. We download the data set from and deploy the same pre-processing suggested in (Romano et al., 2019). Specifically, we selected \(d=150\) features in total, where 75 of them are chosen because they are already reported as important,6. and the rest 75 are those who have the most frequently occurring mutations. After discarding samples that have missing values, we have 1555 examples with 150 features each.1.2 H.2 Semi-synthetic experiments with a non-linear modelHere we provide the complementary results of the semi-synthetic non-linear data experiments, discussed in Sect. 5.2. Figure 20 presents the empirical power and FDR evaluated over 20 realizations of the data set (random train/test splits of the data), using lasso, elastic net and neural network as the base models. In addition to the discussion in Sect. 5.2, it can be seen that the improvement of the neural network is consistent but less significant than those of the linear predictive models (lasso and elastic net). However, although we misspecify the distribution \(P_X\), the empirical FDR is below the nominal level \(q=0.2\) in the neural network as well as lasso and elastic net.Fig. 20Semi-synthetic experiments with real HIV mutation features and a simulated response that follows a non-linear model with varying signal strength c. Empirical power and FDR with \(q= 0.2\) are evaluated over 20 random train/test splits of the data. The other details are as in Fig. 3In addition to the non-linear semi-synthetic experiments described in Sect. 5.2 of the main manuscript, in this section we perform variable selection with real human immunodeficiency virus mutation features and a simulated response that follows a linear model with varying signal strength c. That is, we simulate Y as described in (6) such that \(y \mid X\) follows M1, while treating the real features X as fixed. The results are presented in Fig. 21. As demonstrated, the FDR obtained by all methods is below the target level \(q=0.2\), indicating the robustness of the selection procedure to model misspecification. (Recall that we approximate the unknown \(P_X\) as a multivariate Gaussian whose parameters are estimated from the whole data.) In terms of power, our MRD methods outperform the base predictive models. This is in line with the overall tendency of the experiments presented in this paper, and, in particular, with the ones the correspond to the non-linear semi-synthetic case.Fig. 21Semi-synthetic experiments with real human immunodeficiency virus mutation features and a simulated response that follows a linear model with varying signal strength c. Empirical power and FDR with \(q= 0.2\) are evaluated over 50 random train/test splits of the data. The other details are as in Fig. 3Herein, we extend the experiments from the previous sub-section and study the performance of our method as a function of the sample size. We follow the linear semi-synthetic setting described in Section H.3 with a fixed signal strength \(c=3\), however sample data sets of different size by randomly choosing a subset of \(n