Harvard Data Science Review • 1.1

Machine Learning with Statistical Imputation for Predicting Drug Approvals: Supplementary Materials

Andrew W. Lo, Kien Wei Siah, Chi Heem Wong

Published on: Jun 22, 2019

A Data pre-processing

We construct our datasets from two Informa® databases: *Pharmaprojects* and *Trialtrove*, two separate relational databases organized by largely different ontologies. We extract drug-specific features and drug-indication development status from *Pharmaprojects*, and clinical trial features from *Trialtrove*. We merge the databases through keys provided separately by Informa®.

Pharmaprojects was created earlier than *Trialtrove*, and thus the disease coverage for clinical trials is not as extensive. We start the merging process by first identifying all drug-indication pairs in *Pharmaprojects*. Subsequently, we drop pairs that do not have any trials recorded in *Trialtrove*. As highlighted in Section 2 Materials and methods, profiles in *Pharmaprojects* and *Trialtrove* are fraught with missingness. Therefore, we impose several filters when constructing the datasets to ensure that all instances collected are usable for analysis.

Table 1 summarizes the steps in the filter. We note that the drug, indication, and trial relationships in the constructed datasets are surjective and non-injective: different drugs may target the same indication, and some trials may involve multiple drug-indication pairs. This is logical because it is common that drugs treat multiple diseases, multiple drugs treat a specific disease, or trials involve two or more related primary investigational drugs. To provide some intuition for the size of these databases, we summarize, in Figure 1 and Figure 2 (for P2APP and P3APP respectively), how the number of drug-indication pairs and clinical trials change as we perform the filters.

We extract drug compound attributes and clinical trial characteristics from *Pharmaprojects* and *Trialtrove*, respectively (see Section 1 Data and Table 2). In addition to features readily available in the databases, we create an augmented set of variables capturing sponsor track record and investigator experience. We quantify the track record of sponsors of a specific trial by their success in developing other drugs, using the number of prior approved and failed drug-indication developments; and in past trials for phases 1, 2, and 3 separately, using the total number of trials sponsored, the number of trials sponsored with positive and negative results, and the number of trials sponsored to completion and termination. We use the end date of the last trial of the drug-indication pair under consideration as the cutoff for considering prior experience. This is because the last end date will be the time of prediction. We abstract investigator experience in the same manner. Lastly, we construct a binary drug-indication pair feature, whether the drug has been approved for another indication before. Similarly, we use the end date of the last trial as cutoff for considering prior approval. In total, our datasets have 31 drug-related features and 113 trial-related features.

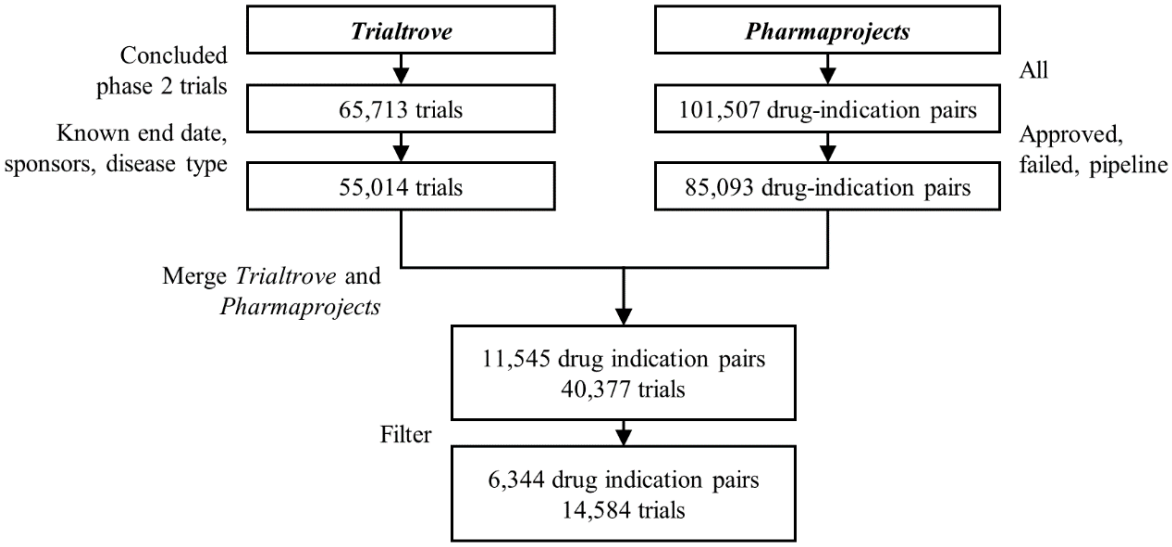

Figure 1. P2APP data filtering.

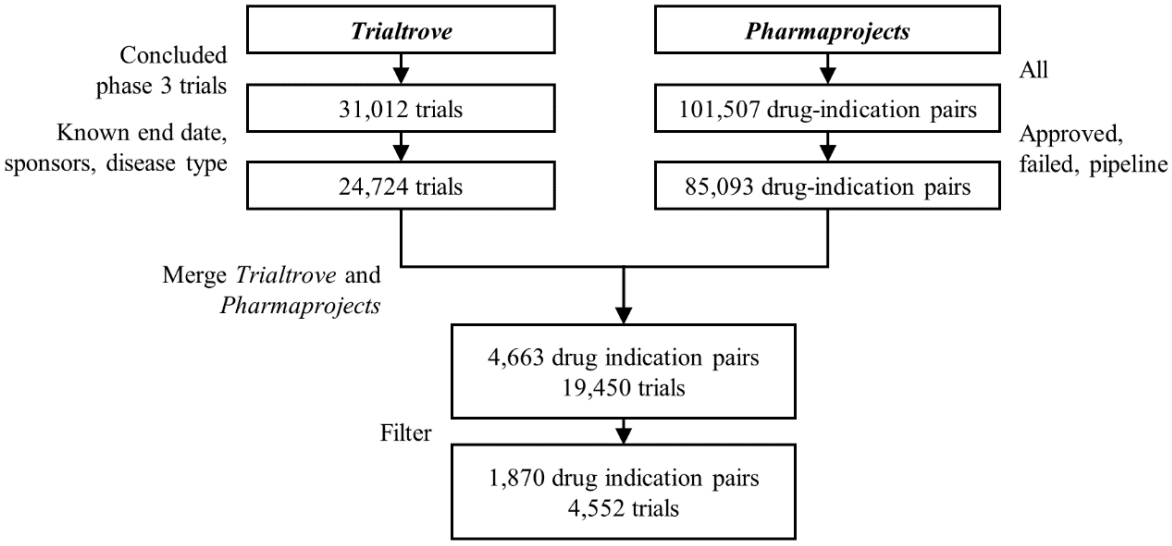

Figure 2. P3APP data filtering.

	Rationale
Drug-indication Pairs in *Pharmaprojects*	
Trials observed in *Trialtrove* (phase 2 for P2APP; phase 3 for P3APP)	We exclude pairs for which we do not observe any trials in *Trialtrove*.
Known approval date (if approved) Approval dates are not available directly in *Pharmaprojects*. They are embedded within text blocks. We mine these text blocks (combination of heuristics and manual extraction) to extract the dates.	We define the approval date as the earliest date a drug-indication pair was approved in any market. We need these dates to create an augmented set of variables capturing sponsors and investigators experience, and also to perform time-series analysis.
Known failure date (if failed)	Failure dates are not directly available in *Pharmaprojects*. We define failure date as one year after the end-date of the last phase 2 or phase 3 trial (if any), whichever is latest.
Clinical Trials in *Trialtrove*	
Phase 2 for P2APP; phase 3 for P3APP	We are interested in predicting approvals using trial features.
Known end date	We need these dates to perform time series analysis. For approved drug-indication pairs in P2APP and P3APP, we compare the trial end date with the corresponding approval date to filter out post-approval trials. These trials may be for supplemental new drug applications (e.g., modified dosage) that are irrelevant to our analysis.
Known sponsors and disease types	Trials not tagged with sponsor/disease types are typically out of *Trialtrove* commercial coverage and are not maintained.

Table 1. Filters for creating datasets.

	Examples	Categories
Drug Features		
Route	Inhaled; Injectable; Oral; Topical	4
Origin	Biological, protein, antibody; Biological, protein, recombinant; Chemical, synthetic	3
Medium	Capsule, hard; Capsule, soft; Powder; Solution; Suspension; Tablet	6
Biological target family	Cytokine/Growth factor; Enzyme; Ion channel; Receptor; Transporter	5
Pharmacological target family	5 Hydroxytryptamine receptor antagonist; Angiogenesis inhibitor; Apoptosis stimulant; Cell cycle inhibitor; DNA inhibitor; DNA synthesis inhibitor; Growth factor receptor antagonist; Immunostimulant; Immunosuppressant; Ion channel antagonist; Protein kinase inhibitor	11
Drug-indication development status	True; false	2
Prior approval of drug for another indication	Approved; failed	2
Trial Features		
Duration	Integer	1
Study design	Active comparator; Cross over; Dose response; Double blind/blinded; Efficacy; Multiple arm; Non-inferiority; Open label; Pharmacodynamics; Pharmacokinetics; Placebo control; Randomized; Safety; Single arm	14
Sponsor type	Academic; Cooperative Group; Government; Industry, all other pharma; Industry, Top 20 Pharma	5
Therapeutic area	Autoimmune/Inflammation; Cardiovascular; CNS; Infectious Disease; Metabolic/Endocrinology; Oncology	6
Trial status	Completed; terminated	2
Trial outcome	Completed, Negative outcome/primary endpoint(s) not met; Completed, Outcome indeterminate; Completed, Positive outcome/primary endpoint(s) met; Terminated, Business decision - Other; Terminated, Business decision - Pipeline reprioritization; Terminated, Lack of efficacy; Terminated, Poor enrollment; Terminated, Safety/adverse effects	8
Target accrual	Integer	1
Actual accrual	Integer	1
Locations	Argentina; Australia; Austria; Belgium; Brazil; Bulgaria; Canada; Chile; Czech Republic; Denmark; Europe; Finland; France; Germany; Hungary; India; Israel; Italy; Japan; Mexico; Netherlands; New Zealand; Peru; Poland; Romania; Russia; Slovakia; South Africa; South Korea; Spain; Sweden; Switzerland; Taiwan; Ukraine; United Kingdom; United States	36
Number of identified sites	Integer	1
Biomarker involvement	Biomarker/Efficacy; Biomarker/Toxicity; PGX - Biomarker Identification/Evaluation; PGX - pathogen; PGX - Patient Preselection/Stratification	5
Sponsor track record	Number of prior approved drug-indication pairs; Number of prior failed pairs; Total number of phase 1 trials sponsored; Number of phase 1 trials with positive results; Number of phase 1 trials with negative results; Number of completed phase 1 trials; Number of terminated phase 1 trials; Total number of phase 2 trials sponsored; Number of phase 2 trials with positive results; Number of phase 2 trials with negative results; Number of completed phase 2 trials; Number of terminated phase 2 trials; Total number of phase 3 trials sponsored; Number of phase 3 trials with positive results; Number of phase 3 trials with negative results; Number of completed phase 3 trials; Number of terminated phase 3 trials	17
Investigator experience	Refer to sponsor track record	17

Table 2. Examples of features extracted from *Pharmaprojects* and *Trialtrove*. After transforming multi-label parent features into binary child features (1 or 0), there were over 3,000 drug and trial categories in total. However, not all are useful for our analysis. For instance, trials rarely take place in Nepal, so the corresponding location feature rarely appears. Thus, this feature is unlikely to have meaningful associations with success. We remove these near zero variance factors. Also, we standardize continuous variables prior to all experiments.

B Missing data definitions

Missing data may be generally classified into three categories (Rubin, 1976): missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR). MCAR holds when data is missing for reasons entirely unrelated to the data, that is, when the probability of missingness is the same for every data point. MAR applies when data missingness can be fully accounted for by observed variables, i.e., when the probability of missingness is the same when conditioned on groups in the observed data. Finally, MNAR comes in when neither MCAR nor MAR is appropriate, when the probability of missingness is dependent on the value of the unobserved variable, or is unknown (Van Buuren, 2012).

For a more precise definition, let Y denote a $n \times p$ data matrix (with elements y_{ij}) where the n rows represent samples and the p columns represent variables. We further partition the observed part of Y as Y_{obs} and the missing part of Y_{mis}, so collectively $Y = (Y_{obs}, Y_{mis})$. Next, let R be a $n \times p$ response indicator matrix where elements $r_{ij} = 0$ if the corresponding element y_{ij} is missing and $r_{ij} = 1$ if y_{ij} is observed. The distribution of R, known as the missing data model/missingness mechanism, may be written generally as $P(R|Y_{obs}, Y_{mis}, \xi)$. R is related in some way to the data Y and is described by some unknown parameters ξ. The missingness is said to be MCAR if $P(R|Y_{obs}, Y_{mis}, \xi) = P(R|\xi)$. This means that the probability of missingness is totally unrelated to the data. The missingness is said to be MAR if $P(R|Y_{obs}, Y_{mis}, \xi) = P(R|Y_{obs}, \xi)$. This means that the missingness does not depend on the values of the missing data when conditioned on the observed data. Finally, the missingness is said to be MNAR if the expression $P(R|Y_{obs}, Y_{mis}, \xi)$ cannot be simplified, i.e., the probability of missingness depends on the unobserved underlying values of the missing data and/or of other observed variables.

Now, we let the distribution of Y, which is the data model we are interested in, be described by some parameters θ. The missingness mechanism can be further described as ignorable under two conditions (Little & Rubin, 2014): First, the missingness must be MAR. Second, the parameters θ and ξ must be distinct[1]. In many situations, the second condition is reasonable because knowing θ will provide little information about ξ and vice versa (Schafer, 1997). In general, the first MAR requirement is considered to be the more important condition. When ignorability holds, Rubin (2004) showed that $P(Y_{mis}|Y_{obs}, R) = P(Y_{mis}|Y_{obs})$. This implies that the distribution of the data is independent of the missing data model, and is identical in both the observed and unobserved groups (Van Buuren, 2012):

$$P(Y|Y_{obs}, R = 1) = P(Y|Y_{obs}, R = 0)$$

In this case, we can model the conditional distribution $P(Y|Y_{obs}, R{=}1)$ from the observed data, and use it to draw imputations for the missing data. In other words, the missing data model R is ignored and not modeled. If the missingness is nonignorable, then (1) does not hold, and the distributions are not

equivalent. When this happens, we need to estimate the missingness mechanism, and incorporate it into the imputation model.

C Statistical imputation methods

Listwise deletion

In listwise deletion, we discard all observations with missing data, in which case there is no imputation. This method is generally not recommended because it is valid only under strict MCAR conditions, which rarely hold in practice. Nevertheless, we can use this as comparison against other methods.

Unconditional mean imputation

In unconditional mean imputation, we fill in the missing values of a variable with the mean/mode of the observed cases of that variable. This method is also highly discouraged because it distorts the data distribution by reducing variability and undermining relationships between variables. In this study, we implement two variants: mean/mode and median/mode imputation.

k-Nearest neighbor imputation

In *k*NN imputation, given an instance with missing values, we select the k most similar cases that do not have missing values in the features to be imputed. As the name suggests, the replacements for the missing values are chosen from these *k* nearest neighbors. In this article, we use the Gower distance for mixed variables[2] and explore 5 and 10 nearest neighbors. For each missing value to be imputed, we use the median/mode of the corresponding feature of the k closest neighbors as the imputation.

Multiple imputation

Multiple imputation (MI) is a principled missing data method that involves three steps: imputation, analysis, and pooling. In the first step, we specify an imputation model for each incomplete variable in the form of a conditional distribution, that is, missing data conditioned on the observed data. Then we draw multiple plausible values for each missing data point according to the specified variable models, creating multiple imputed datasets from one incomplete dataset. In this study, we specify linear regression models for continuous variables and logistic regression models for nominal/categorical

variables. In the second step, we analyze each imputed dataset individually using standard statistical procedures. Finally, in the third step, we pool the estimates obtained from the multiple individual analyses (e.g., probability predictions, regression coefficients) using Rubin's rules (Rubin, 2004) to yield a single estimate. See Supplementary Materials D for more details on MI.

Decision tree algorithm

Decision trees are commonly used as predictive models. In contrast to most machine learning algorithms, some decision tree algorithms can handle missing values internally without the need for imputation. In this paper, we focus on the C5.0 algorithm,[3] a tree-based model developed by Quinlan (1998). It uses entropy as the node impurity measure. When considering a variable for a split, C5.0 uses only examples for which that variable is not missing to calculate the node impurity. When an instance sent down C5.0 encounters a split variable for which it has a missing value, it is split into the branches fractionally, according to the split proportion of the observed instances.

D Notes on multiple imputation

Multiple imputation (MI) is a principled missing data method that can provide valid statistical inferences when missingness is ignorable. It involves three steps: imputation, analysis and pooling (see Figure 3).

Imputation

Under MI, we draw multiple plausible values for each missing data point, thus creating multiple imputed datasets from one incomplete dataset. There are different strategies for multivariate multiple imputation. In this paper, we focus on Fully Conditional Specification (FCS), specifically the Multivariate Imputation by Chained Equations (MICE) algorithm[4] (Buuren & Groothuis-Oudshoorn, 2011). In MICE, we first specify an imputation model for each incomplete variable in the form of conditional distributions, missing data conditioned on the observed data. The algorithm starts with simple random draws from the observed data and imputes the incomplete data in an iterative variable-by-variable fashion according to the specified variable models. Each iteration entails one cycle through all the incomplete variables (see Figure 4). The number of iterations should be set such that convergence is reached. This is typically checked by monitoring the means of imputed values and/or the values of regression coefficients and making sure they are stable over the iterations. In practice, a small number of iterations appears to be sufficient, from 10 to 20. Multiple imputed datasets can be generated by running MICE in parallel the desired number of times.

In this study, we specify linear regression models for incomplete continuous variables and logistic regression models for incomplete nominal variables. We monitor convergence by computing the mean/mode of the imputed values and making sure that they are stable over iterations. 20 iterations appear to be sufficient.

Analysis

The analysis after a single imputation is straightforward: We apply any standard, complete-data statistical methods and end up with one set of results. In MI, we have multiple imputed compete datasets. After analyzing them individually using standard statistical procedures, we end up with multiple sets of results. The differences between the sets represent the uncertainty due to the missing data. The pooling step describes how we can combine these sets of results into a single set.

Pooling

In this step, we pool the estimates obtained from multiple individual analyses using Rubin's rules (Rubin, 2004) to yield a single estimate. Let Q be a column vector of the estimands of interest, $\tilde{Q}$ be its estimate, m be the number of imputed datasets, and $\tilde{Q}_l$ be the estimate of the l^{th} repeated analysis. The combined estimate is given by $\overline{Q} = \frac{1}{m}\sum_{l=1}^{m}\widetilde{Q}_l$. Estimates that can be combined using Rubin's rules include means, regression coefficients and probability predictions.

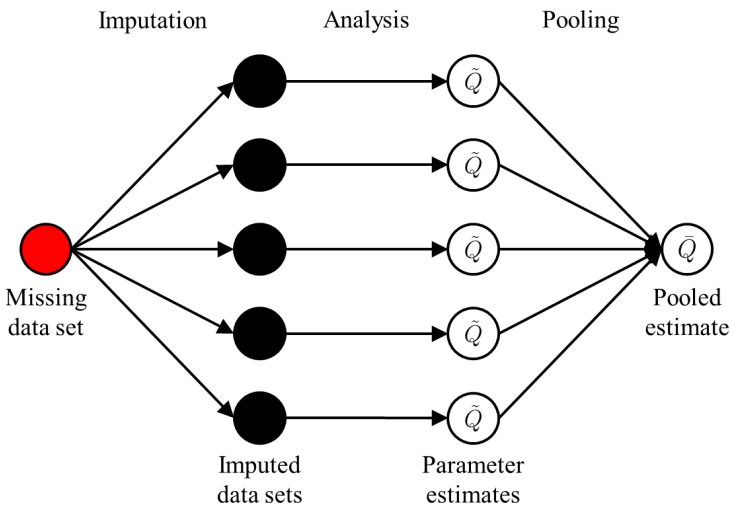

Figure 3. Multiple imputation.

Algorithm: Multivariate Imputation by Chained Equations

Define Y as a $n \times p$ data matrix where rows represent samples and columns represent variables.
Data: Incomplete dataset $Y = \left(Y^{obs}, Y^{mis}\right)$
Result: Incomplete dataset $Y^T = \left(Y^{obs}, Y^{mis,T}\right)$ at iteration T
Define Y_j as the j^{th} feature column of Y where $Y_j = \left(Y_j^{obs}, Y_j^{mis}\right)$
for $j \leftarrow 1$ to p do
| imputation model for incomplete variable $Y_j \leftarrow P\left(Y_j | Y_{-j}, \theta_j\right)$
∟ starting imputations $Y_j^{mis,0} \leftarrow$ draws from Y_j^{obs}
Define $Y_{-j}^t = \left(Y_1^t, Y_2^t, \cdots, Y_{j-1}^t, Y_{j+1}^{t-1}, \cdots, Y_{p-1}^{t-1}, Y_p^{t-1}\right)$ where Y_j^t is the j^{th} feature at iteration t
for $t \leftarrow 1$ to T do
| for $j \leftarrow 1$ to p do
| | $\theta_j^t \leftarrow$ draw from posterior $P\left(\theta_j | Y_j^{obs}, Y_{-j}^t\right)$
∟ ∟ $Y_j^{mis,t} \leftarrow$ draws from posterior predictive $P\left(Y_j^{mis} | Y_{-j}^t, \theta_j^t\right)$
return Y^T

Figure 4. Pseudo-code for Multivariate Imputation by Chained Equations (Van Buuren, 2012).

E Simulation of listwise deletion versus imputation

We design an experiment to study the effects of imputation and verify that imputation indeed offers an improvement over complete-case analysis. First, we create a gold-standard dataset by taking complete cases of the P2APP dataset[5] (see Table 3). Next, we randomly split the drug-indication pairs from the gold-standard dataset into a training set (70%) and a testing set (30%).

To simulate the missingness present in the original dataset, we introduce missingness in the gold-standard training and testing sets based on our MAR assumption and the missingness patterns observed in the P2APP dataset. When making them MAR, we ensure that the proportions of drugs and trials with fully observed features (i.e., complete cases) are consistent with those in the parent dataset (see Supplementary Materials F for a description).

We must be cautious relying on the MAR testing set for model validation. Results may not accurately capture whether a classifier has learned the true underlying relationship between the features and the outcome. To illustrate, suppose that drug-indication pairs have only one binary feature ("0" or "1") that is unrelated to approval/failure. Thus, no classifier can do better than random guessing (0.5 AUC). Now, assume that we have MAR in the dataset: failed pairs are more likely to have missing values due to the data collection process, unrelated to the binary feature. Suppose that we impute all the missing values with 1. Intuitively, we know that this is a poor imputation method because it distorts the feature distribution of failed pairs, and it reduces the variability in the data. However, this is seemingly a "good" method because it allows the AUC of a classifier on this imputed dataset to exceed 0.5. That is, we can identify a disproportionate number of failures by guessing all pairs with feature value 1 as

failures. The classifier has learned a nonexistent relationship introduced by the imputations. By predicting all 1s as failures, the classifier is implicitly exploiting its MAR-ness.

Some may argue that it is acceptable to use missingness as a signal. Unfortunately, this is inappropriate in our case, because the MAR nature of the dataset on hand is merely an artifact of data collection that would not be present during actual testing. MAR was introduced to the data due to the backfilling of information over time[6]. We believe that missingness in current test cases, e.g., drug-indication pairs currently in the pipeline, is more MCAR-like in nature because no backfilling has been performed. For example, immediately after phase 2 testing, pairs that go on to be approved are equally likely to have missing information as pairs that go on to be terminated. Clearly, missingness will not be a useful predictive factor. A classifier that relies heavily on the missingness in the dataset will fail miserably when put into production.

It is difficult to assess how good a classifier really is from the performance on a MAR testing set. Therefore, we create an additional testing set (the "MCAR testing set") in which we introduce missingness based on patterns observed in pipeline drug-indication pairs in the P2APP dataset (see Supplementary Materials F for a description). Because the drugs were still in development at the time of snapshot of the databases, they are likely to be less affected by backfilling. Consequently, the AUC on the MCAR testing set will be more reflective of a classifier's real performance. We also use the gold-standard testing sets for evaluation. These two testing sets serve as a control for the backfilling artifact in the data collection process. They can help to identify non-ideal imputation methods: poor imputation methods tend to distort the data distribution and undermine relationships between variables. This noise makes it more difficult for classifiers to learn the true underlying patterns in the data. These classifiers will perform poorly on the gold-standard and MCAR testing sets[7]. On the other hand, applying imputation methods that are capable of preserving the data distribution will make it easier for classifiers to capture useful relationships in the data. These classifiers will perform well on the gold-standard and MCAR testing sets.

We have two training sets (gold-standard and MAR) and three testing sets (gold-standard, MAR, and MCAR) (see Figure 5). We use five different missing data approaches, as described in Supplementary Materials C, to generate multiple complete training sets from the MAR training set. Subsequently, we use each imputed training set to build six different predictive models (PLR, RF, NN, GBT, SVM, and C5.0) according to the methodology outlined in Section 2 Materials and methods. We use ten-fold cross-validation to select the hyper-parameters for each model. In addition to the imputed MAR training sets, we use the gold-standard training set to train gold-standard classifiers: the models that would have been built if the data was complete. We impute the MAR and MCAR testing sets in a similar fashion as the training sets, and evaluate the AUC performance of all classifiers on the imputed and gold-standard testing sets. We repeat the entire procedure of introducing MAR and MCAR in the

dataset, imputing missingness, training models and validating performance 100 times for robustness. In addition to the AUC, we compute the biasness of the imputed values in the imputed training and testing sets with respect to their gold-standard counterparts. This is a measure of accuracy of each imputation method. Finally, we use the results from the gold-standard, MAR and MCAR testing sets as basis to select an imputation method and machine learning algorithm combination most suitable to the dataset on hand.

Table 5 summarizes the results. Since the training and testing sets are fixed, using the same drug-indication pairs for all methods, direct comparison across different missing data techniques and machine learning algorithms is possible. Each row corresponds to a different missing data technique used to process the training and testing sets in the experiments. Each column group corresponds to a different type of missingness introduced in the testing sets. For all six machine learning algorithms, we find that gold-standard classifiers consistently outperform their complete-case analysis and imputation counterparts. This is logical because useful information is invariably lost when we intentionally introduce missingness in the datasets. In contrast, complete-case analysis often leads to inferior performance. The AUCs of classifiers trained on complete-cases training sets are on average 0.04 less than those trained on imputed training sets. As expected, complete cases are ill suited for MAR data. This supports our conjecture that the use of imputation has allowed predictive models to learn useful patterns that would otherwise be lost from discarding incomplete data.

When comparing across rows, we observe that the different imputation techniques are not equally effective. In terms of imputation quality, MI and mean/mode give the most inaccurate imputations while nearest neighbors recovers data best for both continuous and nominal variables (see Table 4).

To better visualize each imputation method, Figure 6 plots the distributions of the trial feature of actual accrual, a continuous variable, in the gold-standard, complete-cases and imputed MAR training sets of one iteration. It is evident that mean and median imputations have distorted the variable distribution, introducing previously absent peaks at the observed mean and median respectively. In contrast, MI and nearest neighbors imputation managed to preserve the general shape of the variable distribution without introducing anomalous peaks.

We believe that the noise introduced by mean and median imputations have an adverse impact on a classifier's learning process. These effects may not be obvious from the AUC of the MAR testing sets. Indeed, for all six machine learning algorithms, we observe that mean and median imputations give the highest AUCs for the MAR testing sets. However, the trend is reversed when we look at the gold-standard and MCAR testing sets. Classifiers trained on mean or median imputation performed the worst of all imputation methods on these testing sets, implying that the noise introduced by the distortions must have hindered the machine learning algorithms from fully capturing the underlying relationships in the data. It will therefore be prudent to avoid this imputation approach.

Overall, we find *k*NN imputation to be most suitable to the dataset[8]. It provides the least biased imputations among all missing data methods. More importantly, classifiers built on kNN-imputed training sets give the highest AUCs for the gold-standard testing set for all machine learning models explored. By preserving the original data distribution while filling in missing values, *k*NN imputation has allowed classifiers to learn underlying patterns more effectively. In particular, the combination of 5NN with RF gives the one of the highest gold-standard (0.805) and MCAR (0.780) testing set AUCs. This may be attributed to the fact that RF is a nonlinear model, and thus it is able to better capture the complex interactions between the features and regulatory approval than PLR, a linear model. We focus on the 5NN-RF combination in our analyses, since it appears that this pair is most compatible with our datasets.

	Counts				
	Drug-indication Pairs	**Phase 2 Trials**	**Unique Drugs**	**Unique Indications**	**Unique Phase 2 Trials**
Success	166	341	152	83	337
Failure	812	1,672	503	158	1,549
Total	**978**	**2,013**	**623**	**171**	**1,872**

Table 3. Sample size of the gold-standard dataset (derived from complete cases of P2APP).

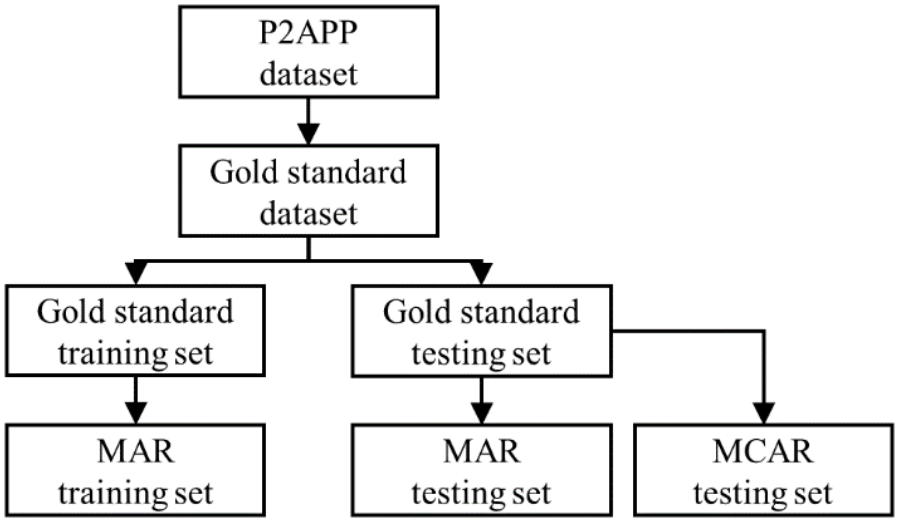

Figure 5. Datasets created in the experiment.

	MAR Training Set		MAR Testing Set		MCAR Testing Set	
	Bias[a]	Wrongly Imputed[b]	Bias[a]	Wrongly Imputed[b]	Bias[a]	Wrongly Imputed[b]
	%	%	%	%	%	%
Mean/mode	234.6	23.0	236.2	23.3	274.2	22.2
Median/mode	115.5	23.0	116.1	23.3	128.7	22.2
5NN	95.4	22.7	94.9	22.0	96.2	21.8
10NN	87.3	21.7	87.9	21.2	90.2	21.0
MI (m=1)	262.0	25.3	268.9	27.9	323.0	26.9
MI (m=10)	260.9	25.3	269.0	27.9	322.7	26.7

[a] Average percentage bias of imputed continuous variables. We first find the sum of the absolute percentage difference between imputed values that are continuous and their corresponding gold-standard values (gold-standard values as denominator), averaged over the total number of missing values that are continuous. We then take the mean over 100 iterations.

[b] Percentage of nominal variables that were wrongly imputed. We first find the number of imputed categorical values that differ from their corresponding gold-standard values, averaged over the total number of missing values that are categorical. Next, we take the mean over 100 iterations.

Table 4. Biasness of imputations with respect to gold standard. Abbreviations: Abs: absolute.

	Testing Set AUC														
	MAR					MCAR					Gold Standard				
	Avg	Sd	5%	50%	95%	Avg	Sd	5%	50%	95%	Avg	Sd	5%	50%	95%
PLR															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.810	0.028	0.761	0.808	0.853
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.755	0.040	0.683	0.764	0.813
Mean/mode	0.786	0.028	0.746	0.785	0.829	0.751	0.029	0.702	0.753	0.794	0.778	0.031	0.729	0.779	0.823
Median/mode	0.786	0.028	0.745	0.786	0.829	0.751	0.029	0.704	0.753	0.794	0.778	0.031	0.728	0.779	0.824
5NN	0.763	0.032	0.716	0.762	0.814	0.757	0.032	0.707	0.758	0.805	0.786	0.032	0.738	0.787	0.834
10NN	0.774	0.030	0.730	0.773	0.821	0.757	0.032	0.695	0.756	0.802	0.787	0.032	0.739	0.791	0.835
MI (m=1)	0.746	0.035	0.688	0.747	0.804	0.758	0.035	0.705	0.755	0.818	0.781	0.036	0.722	0.777	0.843
MI (m=10)	0.755	0.030	0.705	0.757	0.801	0.766	0.032	0.719	0.764	0.815	0.782	0.031	0.729	0.782	0.831
RF															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.837	0.027	0.793	0.837	0.876
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.764	0.048	0.685	0.772	0.830
Mean/mode	0.794	0.027	0.753	0.794	0.836	0.761	0.030	0.712	0.761	0.809	0.775	0.031	0.726	0.771	0.822
Median/mode	0.793	0.027	0.756	0.793	0.831	0.759	0.030	0.709	0.762	0.808	0.774	0.031	0.723	0.774	0.827
5NN	**0.782**	**0.031**	**0.735**	**0.783**	**0.830**	**0.780**	**0.030**	**0.734**	**0.783**	**0.828**	**0.805**	**0.033**	**0.755**	**0.805**	**0.857**
10NN	0.788	0.029	0.741	0.786	0.833	0.780	0.030	0.729	0.778	0.827	0.802	0.033	0.747	0.805	0.856
MI (m=1)	0.774	0.028	0.732	0.777	0.825	0.782	0.031	0.737	0.779	0.845	0.797	0.033	0.748	0.795	0.853
MI (m=10)	0.782	0.029	0.734	0.781	0.831	0.791	0.029	0.739	0.790	0.835	0.804	0.030	0.751	0.804	0.848
NN															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.800	0.032	0.754	0.799	0.849
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.715	0.043	0.638	0.716	0.779
Mean/mode	0.789	0.030	0.736	0.790	0.835	0.766	0.037	0.709	0.766	0.819	0.790	0.037	0.739	0.789	0.848
Median/mode	0.788	0.030	0.742	0.788	0.835	0.766	0.034	0.711	0.766	0.818	0.789	0.036	0.740	0.792	0.849
5NN	0.776	0.030	0.730	0.776	0.821	0.771	0.035	0.715	0.774	0.823	0.794	0.032	0.743	0.798	0.842
10NN	0.784	0.034	0.724	0.785	0.842	0.773	0.039	0.702	0.776	0.831	0.797	0.036	0.737	0.798	0.851
MI (m=1)	0.753	0.035	0.689	0.758	0.801	0.764	0.037	0.708	0.760	0.820	0.780	0.036	0.719	0.781	0.838
MI (m=10)	0.774	0.028	0.729	0.774	0.816	0.784	0.031	0.725	0.789	0.827	0.795	0.030	0.750	0.795	0.838
GBT															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.820	0.028	0.776	0.821	0.868
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.746	0.050	0.659	0.756	0.816
Mean/mode	0.793	0.029	0.746	0.795	0.839	0.762	0.029	0.716	0.763	0.808	0.781	0.034	0.724	0.784	0.826
Median/mode	0.792	0.030	0.743	0.793	0.832	0.760	0.030	0.708	0.762	0.804	0.778	0.033	0.719	0.783	0.823
5NN	0.780	0.030	0.732	0.779	0.821	0.772	0.032	0.717	0.772	0.822	0.796	0.029	0.737	0.798	0.837
10NN	0.787	0.026	0.747	0.788	0.830	0.773	0.028	0.722	0.773	0.817	0.796	0.028	0.748	0.798	0.838
MI (m=1)	0.763	0.031	0.714	0.762	0.812	0.773	0.031	0.727	0.768	0.820	0.796	0.031	0.747	0.796	0.847
MI (m=10)	0.778	0.029	0.733	0.780	0.822	0.789	0.030	0.739	0.789	0.838	0.804	0.031	0.757	0.803	0.854
SVM															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.785	0.030	0.730	0.786	0.831
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.733	0.053	0.650	0.741	0.795
Mean/mode	0.772	0.032	0.724	0.773	0.820	0.741	0.032	0.686	0.748	0.788	0.766	0.036	0.707	0.771	0.818
Median/mode	0.771	0.029	0.729	0.768	0.817	0.740	0.031	0.683	0.745	0.780	0.764	0.035	0.711	0.771	0.818
5NN	0.751	0.031	0.699	0.748	0.803	0.745	0.034	0.697	0.746	0.800	0.771	0.034	0.722	0.770	0.827
10NN	0.758	0.035	0.688	0.760	0.814	0.745	0.037	0.679	0.749	0.808	0.772	0.037	0.710	0.773	0.825
MI (m=1)	0.731	0.035	0.676	0.732	0.788	0.741	0.033	0.684	0.745	0.790	0.760	0.035	0.696	0.762	0.813
MI (m=10)	0.746	0.030	0.705	0.746	0.797	0.755	0.031	0.707	0.753	0.797	0.768	0.030	0.719	0.764	0.813
C5.0															
Gold Standard	-	-	-	-	-	-	-	-	-	-	0.800	0.033	0.758	0.800	0.844
Complete Cases	-	-	-	-	-	-	-	-	-	-	0.710	0.063	0.585	0.713	0.802
Mean/mode	0.764	0.033	0.711	0.768	0.810	0.734	0.032	0.675	0.737	0.777	0.758	0.039	0.698	0.762	0.816
Median/mode	0.764	0.038	0.708	0.761	0.825	0.735	0.041	0.676	0.736	0.797	0.754	0.043	0.679	0.751	0.823
5NN	0.756	0.036	0.703	0.753	0.816	0.749	0.038	0.695	0.745	0.805	0.772	0.038	0.715	0.772	0.843
10NN	0.759	0.035	0.696	0.762	0.807	0.747	0.037	0.687	0.749	0.799	0.770	0.035	0.710	0.771	0.822
MI (m=1)	0.733	0.038	0.672	0.731	0.795	0.741	0.036	0.680	0.740	0.800	0.758	0.037	0.701	0.754	0.819
MI (m=10)	0.786	0.030	0.738	0.786	0.836	0.793	0.031	0.738	0.797	0.842	0.807	0.031	0.756	0.808	0.857
MAR[a]	0.759	0.037	0.699	0.759	0.811	0.744	0.037	0.685	0.741	0.801	0.761	0.037	0.705	0.757	0.812

[a] For MAR, we leave the missingness as it is and rely on the decision tree algorithm to handle them internally.

Table 5. AUC of different classifiers under different missing data approaches. Abbreviations: Avg: average; Sd: standard deviation; 5%: 5th percentile; 50%: median; 95%: 95th percentile; m: number of imputations generated.

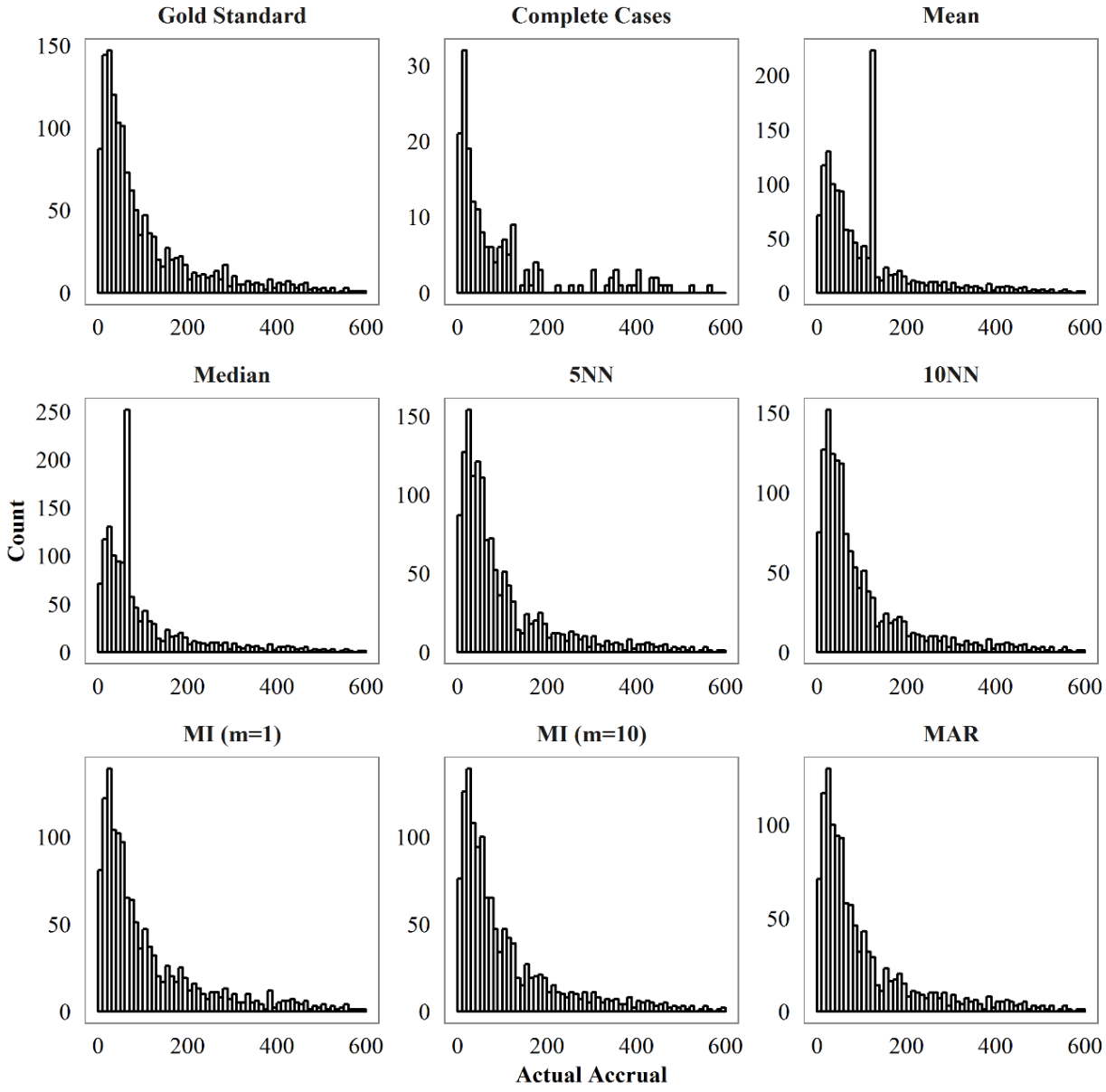

Figure 6. Gold-standard, complete-cases, MAR and imputed distributions of actual accrual in the training set of one of the iterations. The range of actual accrual goes up to 3,000. However, only a small number of samples go beyond 600. Thus, we truncate the histograms at 600 for better visualization. For MAR distribution, we ignore all missing values.

F Making MAR and MCAR Training Sets

We simulate missingness in gold-standard training and testing sets (see Table 3) based on our assumption of MAR and the missingness patterns observed in the P2APP dataset (see Table 6 and

Table 7). For example, 36% of approved drugs in the P2APP dataset have some incomplete drug features. Accordingly, we randomly select 36% of approved drugs in the gold-standard training set and introduce missingness in drug features according to the observed proportions to form the MAR training set, e.g., 6% of these drugs will have missing pharmacological target family values, 76% will have missing biological target family values, and so on. We repeat this process for failed drugs, completed trials, and terminated trials. At the end, we propagate the missing drug and trial features into the training set feature matrix, so that drug-indication pairs for the same drug have the same drug features missing in their feature vectors, and drug-indication pairs with the same trial have the same trial features missing. Conversely, when making the sets MAR, we ensure that the proportions of drugs and trials with fully observed features (i.e., complete-cases) are consistent with that observed in the parent dataset, e.g., 64% of approved drugs in the MAR training set have complete drug features. We repeat this procedure for the gold-standard testing set to form the MAR testing set.

We simulate MCAR in the gold-standard testing set in a similar fashion to form the MCAR testing set. However, here we use unconditional missingness patterns observed in the pipeline dataset (see Table 6 and Table 7), instead of the known outcomes set where backfilling has occurred.

	Missingness[a]		
	Known Outcomes		**Pipeline**
	Success	**Failure**	**Unconditional**
COMPLETE CASES	0.64	0.29	0.46
INCOMPLETE CASES	0.36	0.71	0.54
Route	0.00	0.06	0.04
Pharmacological target family	0.06	0.10	0.17
Biological target family	0.76	0.45	0.63
Medium	0.43	0.86	0.69

[a] Feature missingness with respect to incomplete cases, e.g., 36% of success drugs have some incomplete drug features. 43% of these drugs have missing medium values.

Table 6. Breakdown of missingness in drug features in P2APP with respect to unique drugs (see Section 2 Missing data).

	Missingness[a]		
	Known Outcomes		Pipeline
	Success	Failure	Unconditional
COMPLETE CASES	0.22	0.60	0.44
INCOMPLETE CASES	0.78	0.40	0.56
Number of identified sites	0.13	0.24	0.21
Actual accrual	0.13	0.54	0.18
Duration	0.37	0.13	0.24
Target accrual	0.54	0.21	0.37
Locations	0.02	0.04	0.02
Study design keywords	0.31	0.24	0.13
Trial outcomes	0.93	0.27	0.81

[a] Feature missingness with respect to incomplete cases.

Table 7. Breakdown of missingness in trial features in P2APP with respect to unique trials (see Section 2 Missing data).

G Comparison of general and indication-group specific classifiers

	General Classifier			Specialized Classifiers		
	Train Set	Test Set	AUC [95% CI]	Train Set	Test Set	AUC [95% CI]
All						
2004–2008	1,361	551	0.669 [0.614, 0.725]	-	-	-
2005–2009	1,562	591	0.680 [0.625, 0.735]	-	-	-
2006–2010	1,764	636	0.712 [0.668, 0.755]	-	-	-
2007–2011	1,969	598	0.738 [0.698, 0.777]	-	-	-
2008–2012	2,082	597	0.799 [0.760, 0.837]	-	-	-
2009–2013	2,212	517	0.823 [0.779, 0.867]	-	-	-
2010–2014	2,289	380	0.797 [0.718, 0.876]	-	-	-
Alimentary						
2004–2008	1,361	86	0.494 [0.294, 0.694]	170	86	0.502 [0.310, 0.694]
2005–2009	1,562	93	0.613 [0.440, 0.785]	197	93	0.459 [0.287, 0.630]
2006–2010	1,764	80	0.589 [0.447, 0.731]	237	80	0.491 [0.321, 0.662]
2007–2011	1,969	77	0.707 [0.592, 0.821]	257	77	0.541 [0.396, 0.686]
2008–2012	2,082	67	0.802 [0.694, 0.909]	275	67	0.402 [0.252, 0.553]
2009–2013	2,212	58	0.834 [0.715, 0.954]	279	58	0.610 [0.441, 0.780]
2010–2014	2,289	39	0.670 [0.427, 0.913]	274	39	0.656 [0.414, 0.899]
Cardiovascular						
2004–2008	1,361	39	0.515 [0.313, 0.717]	93	39	0.541 [0.310, 0.771]
2005–2009	1,562	38	0.307 [0.104, 0.509]	105	38	0.452 [0.230, 0.674]
2006–2010	1,764	46	0.613 [0.430, 0.795]	118	46	0.628 [0.449, 0.806]
2007–2011	1,969	37	0.634 [0.396, 0.872]	135	37	0.793 [0.644, 0.942]
2008–2012	2,082	42	0.640 [0.426, 0.853]	137	42	0.621 [0.425, 0.818]
2009–2013	2,212	35	0.360 [0.138, 0.582]	145	35	0.460 [0.272, 0.648]
2010–2014	2,289	19	0.529 [0.000, 1.000]	148	19	0.618 [0.000, 1.000]
Anti-infective						
2004–2008	1,361	46	0.658 [0.502, 0.815]	124	46	0.645 [0.478, 0.812]
2005–2009	1,562	44	0.695 [0.525, 0.866]	146	44	0.707 [0.551, 0.863]
2006–2010	1,764	53	0.733 [0.568, 0.897]	161	53	0.708 [0.552, 0.864]
2007–2011	1,969	44	0.648 [0.479, 0.818]	171	44	0.592 [0.420, 0.763]
2008–2012	2,082	43	0.801 [0.666, 0.936]	165	43	0.815 [0.684, 0.945]
2009–2013	2,212	32	0.658 [0.454, 0.862]	169	32	0.649 [0.435, 0.864]
2010–2014	2,289	18	0.875 [0.708, 1.000]	167	18	0.750 [0.515, 0.985]
Anti-cancer						
2004–2008	1,361	137	0.665 [0.528, 0.803]	456	137	0.683 [0.533, 0.833]
2005–2009	1,562	163	0.739 [0.618, 0.861]	494	163	0.635 [0.512, 0.758]
2006–2010	1,764	188	0.774 [0.702, 0.846]	546	188	0.726 [0.635, 0.816]
2007–2011	1,969	193	0.830 [0.773, 0.887]	618	193	0.746 [0.661, 0.831]
2008–2012	2,082	198	0.805 [0.717, 0.894]	682	198	0.760 [0.665, 0.855]
2009–2013	2,212	177	0.852 [0.783, 0.922]	736	177	0.786 [0.696, 0.876]
2010–2014	2,289	173	0.815 [0.691, 0.938]	791	173	0.803 [0.666, 0.940]
Musculoskeletal						
2004–2008	1,361	35	0.765 [0.597, 0.933]	96	35	0.704 [0.512, 0.896]
2005–2009	1,562	38	0.716 [0.489, 0.944]	109	38	0.674 [0.472, 0.876]
2006–2010	1,764	35	0.634 [0.439, 0.830]	111	35	0.509 [0.276, 0.742]
2007–2011	1,969	37	0.737 [0.571, 0.903]	119	37	0.677 [0.493, 0.860]
2008–2012	2,082	36	0.884 [0.773, 0.995]	127	36	0.683 [0.462, 0.904]
2009–2013	2,212	26	0.792 [0.573, 1.000]	133	26	0.667 [0.429, 0.904]
2010–2014	2,289	19	0.882 [0.724, 1.000]	128	19	0.882 [0.706, 1.000]
Neurological						
2004–2008	1,361	122	0.688 [0.572, 0.803]	211	122	0.768 [0.676, 0.859]
2005–2009	1,562	119	0.612 [0.471, 0.753]	271	119	0.625 [0.501, 0.748]
2006–2010	1,764	125	0.656 [0.532, 0.779]	334	125	0.673 [0.560, 0.787]
2007–2011	1,969	105	0.701 [0.580, 0.822]	375	105	0.649 [0.522, 0.776]
2008–2012	2,082	114	0.806 [0.707, 0.904]	382	114	0.695 [0.586, 0.804]
2009–2013	2,212	87	0.938 [0.857, 1.000]	417	87	0.718 [0.558, 0.879]
2010–2014	2,289	55	0.984 [0.952, 1.000]	408	55	0.860 [0.721, 0.999]
Respiratory						
2004–2008	1,361	34	0.673 [0.418, 0.927]	89	34	0.833 [0.650, 1.000]
2005–2009	1,562	42	0.842 [0.722, 0.962]	104	42	0.825 [0.670, 0.979]
2006–2010	1,764	49	0.797 [0.663, 0.931]	125	49	0.801 [0.644, 0.959]
2007–2011	1,969	36	0.694 [0.513, 0.875]	143	36	0.519 [0.323, 0.715]
2008–2012	2,082	43	0.751 [0.604, 0.899]	149	43	0.692 [0.520, 0.865]
2009–2013	2,212	37	0.827 [0.694, 0.961]	154	37	0.876 [0.764, 0.987]
2010–2014	2,289	23	0.724 [0.365, 1.000]	160	23	0.842 [0.679, 1.000]
Rare Diseases						
2004–2008	1,361	69	0.664 [0.517, 0.811]	212	69	0.521 [0.349, 0.693]
2005–2009	1,562	81	0.627 [0.471, 0.782]	231	81	0.528 [0.368, 0.687]
2006–2010	1,764	108	0.774 [0.666, 0.881]	257	108	0.691 [0.546, 0.836]
2007–2011	1,969	101	0.786 [0.698, 0.874]	303	101	0.680 [0.547, 0.812]
2008–2012	2,082	112	0.787 [0.696, 0.879]	329	112	0.600 [0.469, 0.731]
2009–2013	2,212	90	0.803 [0.702, 0.903]	358	90	0.730 [0.626, 0.834]
2010–2014	2,289	89	0.793 [0.621, 0.965]	391	89	0.779 [0.626, 0.932]

Table 8. Comparison of the general and indication-group specific classifiers for selected indication group in P2APP. We use bootstrapping to determine the 95% CI for AUC. We exclude indication groups with too few samples.

	General Classifier			Specialized Classifiers		
	Train Set	Test Set	AUC [95% CI]	Train Set	Test Set	AUC [95% CI]
			All			
2004–2008	472	196	0.769 [0.704, 0.834]	-	-	-
2005–2009	559	177	0.724 [0.650, 0.798]	-	-	-
2006–2010	604	211	0.738 [0.671, 0.805]	-	-	-
2007–2011	664	174	0.806 [0.740, 0.871]	-	-	-
2008–2012	677	197	0.827 [0.768, 0.886]	-	-	-
2009–2013	740	153	0.868 [0.809, 0.927]	-	-	-
2010–2014	734	110	0.876 [0.811, 0.941]	-	-	-
			Alimentary			
2004–2008	472	65	0.826 [0.651, 1.000]	25	65	0.889 [0.756, 1.000]
2005–2009	559	75	0.683 [0.324, 1.000]	17	75	0.650 [0.331, 0.969]
2006–2010	604	80	0.672 [0.428, 0.915]	30	80	0.651 [0.429, 0.872]
2007–2011	664	91	0.911 [0.786, 1.000]	28	91	0.800 [0.630, 0.970]
2008–2012	677	97	0.786 [0.572, 1.000]	24	97	0.700 [0.469, 0.931]
2009–2013	740	107	0.607 [0.149, 1.000]	18	107	0.786 [0.570, 1.000]
2010–2014	734	99	0.944 [0.850, 1.000]	19	99	0.733 [0.492, 0.975]
			Anti-cancer			
2004–2008	472	95	0.773 [0.618, 0.928]	34	95	0.684 [0.495, 0.874]
2005–2009	559	107	0.740 [0.543, 0.936]	28	107	0.568 [0.345, 0.791]
2006–2010	604	110	0.754 [0.599, 0.910]	50	110	0.630 [0.452, 0.809]
2007–2011	664	132	0.587 [0.333, 0.842]	24	132	0.392 [0.132, 0.651]
2008–2012	677	134	0.793 [0.549, 1.000]	40	134	0.668 [0.457, 0.879]
2009–2013	740	151	0.800 [0.480, 1.000]	29	151	0.775 [0.528, 1.000]
2010–2014	734	153	0.943 [0.842, 1.000]	26	153	0.852 [0.558, 1.000]
			Neurological			
2004–2008	472	118	0.851 [0.753, 0.949]	59	118	0.837 [0.735, 0.939]
2005–2009	559	151	0.782 [0.646, 0.918]	45	151	0.784 [0.649, 0.919]
2006–2010	604	169	0.732 [0.593, 0.871]	52	169	0.759 [0.629, 0.890]
2007–2011	664	180	0.706 [0.532, 0.880]	40	180	0.698 [0.529, 0.867]
2008–2012	677	178	0.765 [0.604, 0.926]	41	178	0.743 [0.586, 0.900]
2009–2013	740	185	0.827 [0.681, 0.973]	31	185	0.805 [0.641, 0.968]
2010–2014	734	166	0.779 [0.567, 0.990]	27	166	0.900 [0.782, 1.000]
			Rare Disease			
2004–2008	472	54	0.711 [0.465, 0.957]	22	54	0.620 [0.364, 0.876]
2005–2009	559	60	0.735 [0.517, 0.952]	23	60	0.606 [0.360, 0.852]
2006–2010	604	66	0.888 [0.747, 1.000]	24	66	0.825 [0.645, 1.000]
2007–2011	664	72	0.838 [0.652, 1.000]	22	72	0.735 [0.520, 0.950]
2008–2012	677	76	0.893 [0.780, 1.000]	34	76	0.700 [0.523, 0.877]
2009–2013	740	94	0.962 [0.899, 1.000]	28	94	0.932 [0.840, 1.000]
2010–2014	734	109	0.908 [0.766, 1.000]	18	109	0.985 [0.942, 1.000]

Table 9. Comparison of the general and indication-group specific classifiers for selected indication group in P3APP. We use bootstrapping to determine the 95% CI for AUC. We exclude indication groups with too few samples.

H Comparison with DiMasi et al. (2015)

The Approved New Drug Index (ANDI) algorithm was proposed by DiMasi et al. (2015) to predict regulatory approval for lead indications of cancer drugs after phase 2 testing. It is composed of a rubric of four factors to score anticancer agents (see Supplementary Materials I). The factors are based on pivotal trial characteristics and disease prevalence. Higher scores correspond to a higher probability of success. In this analysis, we apply ANDI on the oncology samples in the P2APP dataset, analyze its performance, and compare it with our 5NN-RF classifier in Supplementary Materials E.

First, we extract all cancer drugs from P2APP to form an oncology-only dataset. Since ANDI requires complete cases, we drop all examples with missing values in any of the four ANDI factors (see Table 10 for the resulting sample size). From this dataset, we draw a training set of 62 drugs with the same

composition as that used by DiMasi et al. (2015): 40 failures and 22 successes. We set aside the remaining 319 drugs as a held-out testing set.

In replicating the ANDI experiment, we endeavor to follow the original proposed rubric as closely as possible. Unfortunately, two factors in the rubric are not in our dataset. We replace them with surrogate variables, and tune their cutoffs using the training set put aside earlier. The modified rubric is given in Table 11. In order to apply ANDI, we must identify the lead indication of each oncology drug and the pivotal phase 2 trial for that drug-indication pair. However, DiMasi et al. (2015) did not provide clear instructions for identifying lead indications or pivotal trials. In this experiment, we apply heuristics which we felt were most logical. See Supplementary Materials I for details on the proxy variables and heuristics used.

DiMasi et al. (2015) reported an impressive 0.92 AUC for ANDI on a dataset of 62 drugs. However, this figure is based on in-sample/training-set testing, i.e., the algorithm was tested on the dataset on which the scoring rubric itself was derived. Such testing naturally yields excellent results because the four factors and their cutoffs were optimized for the algorithm to do well on the dataset. However, it is nearly impossible to judge whether an algorithm will generalize well without some form of testing on held-out datasets. Unfortunately, such validation was not performed by DiMasi et al. (2015). Furthermore, ANDI was derived from a small sample, making it even more susceptible to overfitting.

For these reasons, it is very likely that the discriminative power of ANDI is actually much lower than that implied by the reported AUC of 0.92. Knowing these issues, we augment the ANDI experiment by including an out-of-sample model validation step, using the 319 drugs set aside as the testing set. This will allow us to determine ANDI's real performance more accurately.

The receiver operating characteristic curves of the original ANDI algorithm as reported in DiMasi et al. (2015) and the modified ANDI on the oncology-only training and testing sets are shown in Figure 7. Similar to the original ANDI, our modified ANDI rubric demonstrates excellent performance on the training set with 0.94 AUC, 95% CI (0.89, 0.99). Unfortunately, this performance does not hold up on the testing set. The modified ANDI managed only 0.69 AUC on new, unseen samples. The large discrepancy between training and testing AUCs is indicative of overfitting. It is apparent that the patterns learned from the small training sample (n=62) do not generalize well, highlighting the importance of proper model validation. We believe the same holds for the original ANDI.

For a direct comparison with our classifiers, we apply the modified ANDI on oncology drugs in the gold-standard testing sets in Supplementary Materials E. Figure 8 summarizes the distributions of the results and compares 5NN-RF with the modified ANDI. On this testing set subsample, we find that our classifier achieves significantly higher AUC than the modified ANDI, with an average improvement of 0.1 in AUC over 100 simulations. We believe that this gain can be attributed to a larger training set with

a wider range of features, a nonlinear model that can capture the complex relationships in the data, and proper model validation methodology.

Lastly, we note that DiMasi et al. (2015) applied complete-case analysis in their study without any characterization of the missingness in their dataset. This is problematic because complete cases are appropriate only under strict MCAR conditions. Violation of these conditions will lead to biased estimates. Since data is rarely MCAR in reality, it is unsurprising that the modified ANDI yields inferior performance. In practice, this limits the applicability of ANDI to only samples with complete information which, given the scattershot nature of reporting in drug development, limits the practical relevance of ANDI.

	Counts				
	Drug-indication Pairs	**Phase 2 Trials**	**Unique Drugs**	**Unique Indications**	**Unique Phase 2 Trials**
Success	71	178	61	28	176
Failure	668	1,345	347	40	1,213
Total	**739**	**1,523**	**381**	**40**	**1,368**

Table 10. Sample size of the oncology-only dataset (derived from P2APP).

	Score		
Factor	**0**	**1**	**2**
Trial outcomes[a]	Terminated, lack of efficacy; completed, negative outcomes or primary endpoints not met	Completed, outcome indeterminate	Terminated, early positive outcomes; completed, positive outcomes or primary endpoints met
Number of patients in pivotal phase 2 trial	≤ 37	38-49	≥ 50
U.S. incidence[a]	> 100,000	10,000-100,000	< 10,000
Phase 2 duration (months)	> 44	21-44	< 21

[a] Surrogate variable.

Table 11. Modified ANDI rubric in this study.

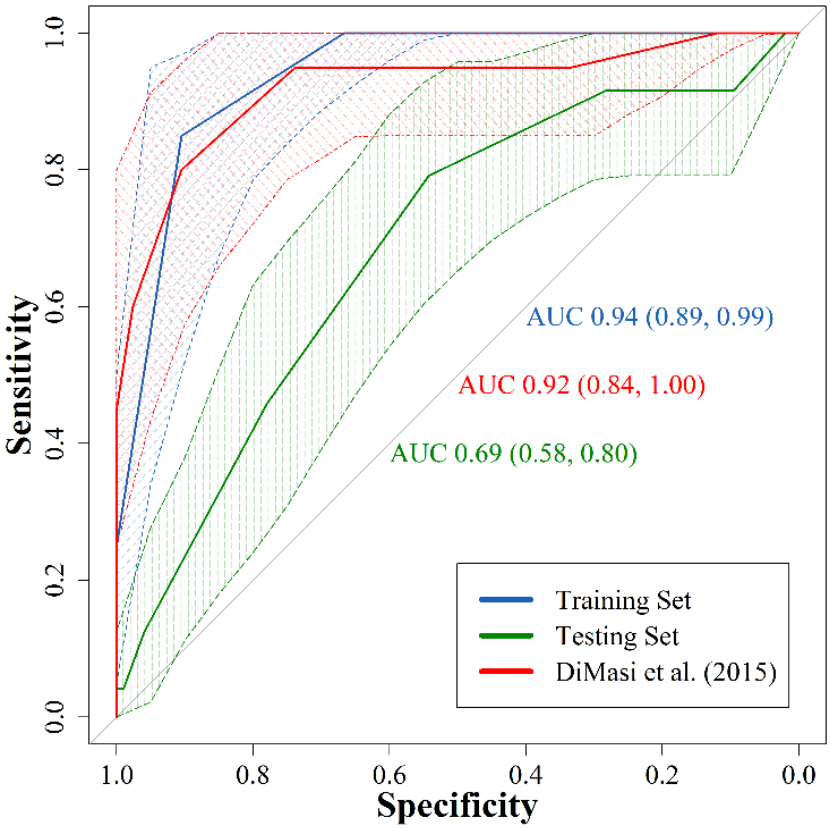

Figure 7. Receiver operating characteristic curves of the original ANDI (as reported in DiMasi et al. (2015)) and the modified ANDI on the oncology-only training and testing sets. We use bootstrapping to determine the 95% CI. We plot the receiver operating characteristic curve of the original ANDI from DiMasi et al. (2015) (red) by using the ANDI scores breakdown provided in the paper. The slight difference in the lower bound of the 95% CI between what we computed (0.84) and what DiMasi et al. reported (0.81) may be explained by randomness in the bootstraps. Abbreviations: ROC: receiver operating characteristic curve.

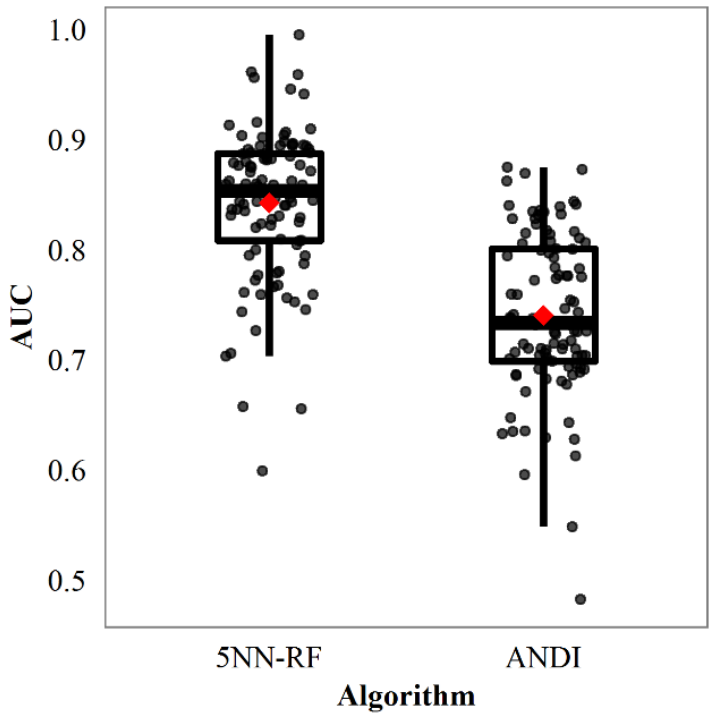

Figure 8. Distributions of AUC of 5NN-RF and the modified ANDI on oncology-only gold-standard testing sets from Supplementary Materials E.

I Modified ANDI

In replicating the ANDI experiment (DiMasi et al., 2015), we endeavor to follow the original proposed design as closely as possible (see Table 12). Unfortunately, two variables in their design are not in our dataset: worldwide prevalence and activity. We replace them with surrogate variables, and tune their cutoffs using the training set placed aside earlier. The modified design is given in Table 11. First, we use US incidence as a proxy for worldwide prevalence. This is because the latter figure is not known accurately for many of the oncology indications in our dataset, while the US incidence is much better documented and more accessible[9]. We determine the cutoffs in a manner similar to DiMasi et al. (2015): a larger incidence has lower scores while a smaller incidence has higher scores. Second, we use the trial outcome (i.e., the results of the trial) as a proxy for activity. We set the cutoffs similarly as in the original rubric: negative results have lower scores, while positive results have higher scores.

In order to apply ANDI, we must identify the lead indication of each oncology drug and the pivotal phase 2 trial for that drug-indication pair. Unfortunately, DiMasi et al. (2015) did not provide clear instructions for identifying lead indications or pivotal trials in their paper[10]. A fair amount of subjectivity appears to have been involved; there was no mention of any concrete criteria in the paper.

This makes it difficult to replicate their study on other datasets. In this experiment, we apply heuristics that we feel are most logical. For drugs with multiple indications, we take the indication with the most phase 2 trials as the lead. We hypothesize that companies will invest in more trials for the designated lead indication. For drug-indication pairs with multiple phase 2 trials, we choose the trial with the largest accrual as the pivotal trial. This is logical, since trials with larger sample size have greater statistical power. They should hold greater weight in the decision of whether to proceed to phase 3 testing. In the event of ties, with the same number of trials or an identical accrual, we randomly select one of the candidates as the lead indication or pivotal trial.

	Score		
Factor	**0**	**1**	**2**
Pivotal phase 2 trial activity	< 3.0% or negative randomized phase 2 trial	3.0-13.8%	< 13.8% or positive randomized phase 2 trial
Number of patients in pivotal phase 2 trial	≤ 37	38-49	≥ 50
No. of patients treated worldwide	> 302,000	50,000-302,000	< 50,000
Phase 2 duration (months)	> 44	21-44	< 21

Table 12. Oncology ANDI proposed by DiMasi et al. (2015).

J Simulation of random splitting versus temporal ordering

We design an experiment to study the effects of any look-ahead bias introduced by splitting drug-indication pairs into training and testing sets randomly without considering the dates of development. First, we sample five-year rolling windows between 2004 and 2014 from the P2APP and P3APP datasets. In Section 3 Predictions over time, we note that each window consists of a training set of drug-indication pairs whose outcomes become finalized within the window, and an out-of-sample, out-of-time testing set of drug-indication pairs that ended phase 2 or phase 3 testing, but are still in the pipeline with undetermined outcomes within the window. Here we disregard the temporal ordering—we aggregate the training and testing sets, and re-split them randomly before applying our machine-learning framework. To allow direct comparison with the time-series approach, we keep the new training and testing sample sizes the same as those in Section 3 Predictions over time. Table 13 summarizes the results.

	Sample Size		AUC [95% CI]	
	Train Set	Test Set	Random Splitting	Temporal Ordering
P2APP				
2004–2008	1,361	551	0.750 [0.703, 0.797]	0.669 [0.614, 0.725]
2005–2009	1,562	591	0.764 [0.720, 0.808]	0.680 [0.625, 0.735]
2006–2010	1,764	636	0.748 [0.703, 0.794]	0.712 [0.668, 0.755]
2007–2011	1,969	598	0.768 [0.727, 0.809]	0.738 [0.698, 0.777]
2008–2012	2,082	597	0.750 [0.705, 0.795]	0.799 [0.760, 0.837]
2009–2013	2,212	517	0.781 [0.732, 0.829]	0.823 [0.779, 0.867]
2010–2014	2,289	380	0.795 [0.732, 0.858]	0.797 [0.718, 0.876]
P3APP				
2004–2008	472	196	0.720 [0.650, 0.790]	0.769 [0.704, 0.834]
2005–2009	559	177	0.748 [0.675, 0.821]	0.724 [0.650, 0.798]
2006–2010	604	211	0.771 [0.707, 0.835]	0.738 [0.671, 0.805]
2007–2011	664	174	0.810 [0.743, 0.877]	0.806 [0.740, 0.871]
2008–2012	677	197	0.805 [0.744, 0.866]	0.827 [0.768, 0.886]
2009–2013	740	153	0.820 [0.754, 0.885]	0.868 [0.809, 0.927]
2010–2014	734	110	0.849 [0.772, 0.925]	0.876 [0.811, 0.941]

Table 13. Comparison of classifiers trained on random splitting and temporal ordering. We use bootstrapping to determine the 95% CI for AUC.

We find that random splitting is indeed susceptible to overoptimistic performance (e.g., first four windows in P2APP). This may be attributed to the presence of future information in the training set, thus leading to look-ahead bias. However, we also observe overpessimistic results in some cases (e.g., last three windows in P3APP). This may occur when useful past information is set aside in the testing set. We believe that historical successes and failures contain valuable insights on the characteristics of high-potential candidates. Consider prediction for a phase 3 drug today. If we know that a drug with similar mechanism of action has been approved before, we should probably assign a higher chance of success to the pipeline drug under consideration. Conversely, if we see termination of drugs with similar mechanism of action in the past, we should lower our expectations for the pipeline drug as well. Under random allocation, the pipeline drug may be set aside in the testing set together with its historical counterpart. This prevents the model from learning from past experiences, which leads to overpessimistic performance.

The use of random splitting may be less than ideal due to the reasons noted above. It is prudent to adhere to the temporal ordering in the dataset when constructing training and testing sets in order to obtain realistic inferences.

K References

Buuren, S., & Groothuis-Oudshoorn, K. (2011). mice: Multivariate imputation by chained equations in R. *Journal of Statistical Software*, 45(3).

DiMasi, J. A., Hermann, J. C., Twyman, K., Kondru, R. K., Stergiopoulos, S., Getz, K. A., & Rackoff, W. (2015). A tool for predicting regulatory approval after phase II testing of new oncology compounds. *Clinical Pharmacology & Therapeutics*, 98(5), 506-513.

Kuhn, M., Weston, S., Coulter, N., & Quinlan, R. (2014). C50: C5. 0 decision trees and rule-based models. R package version 0.1. 0-21.

Little, R. J., & Rubin, D. B. (2014). *Statistical analysis with missing data*. John Wiley & Sons.

Quinlan, R. (1998). C5. 0: An informal tutorial.

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, 63(3), 581-592.

Rubin, D. B. (2004). *Multiple imputation for nonresponse in surveys*. John Wiley & Sons.

Schafer, J. L. (1997). *Analysis of incomplete multivariate data*. CRC Press.

Templ, M., Alfons, A., Kowarik, A., & Prantner, B. (2015). VIM: visualization and imputation of missing values. R package version, 4.4.1.

Van Buuren, S. (2012). *Flexible imputation of missing data*. CRC Press.

Footnotes

1. θ and ξ should be a priori independent where $P(\theta, \xi)$ factors into $P(\theta)P(\xi)$ (Little & Rubin, 2014). ↩
2. Implemented in R, VIM package (Templ, Alfons, Kowarik, & Prantner, 2015). ↩
3. Implemented in R, C50 package (Kuhn, Weston, Coulter, & Quinlan, 2014). ↩
4. Implemented in R, MICE package (Buuren & Groothuis-Oudshoorn, 2011). Van Buuren provides a comprehensive guide to MICE in Van Buuren (2012). ↩
5. We exclude pipeline drug-indication pairs in this analysis because their outcomes are unknown. ↩
6. This occurs due to a combination of reasons—some drug characteristics (e.g., mechanism of action) only become clear as the study progresses to higher phases; poor reporting practices. ↩
7. Returning to the above binary feature example, if we had tested the classifier on a gold-standard testing set, we would realize that it did not learn any useful patterns. ↩
8. Note that the MI (m=10)-RF and MI (m=10)-C5.0 combinations yielded slightly better performances than *k*NN-RF. However, we excluded MI (m=10) from consideration because the improvement is only marginal while the imputation and analysis processes are much more time

consuming, since we have ten imputed datasets in MI (m=10). Furthermore, the imputation method does not converge well (or at all) for smaller datasets. This poses an issue for the time series analysis in Section 3 Predictions over time. In contrast, *k*NN imputation is relatively straightforward to implement and more stable. ↩

9. Sources include the American Cancer Society and the National Cancer Institute Surveillance, Epidemiology and End Results Program. ↩

10. Attention was focused on what they "determined to be the lead cancer indication pursued", and they "identified what appeared to be the phase II trial that was most pivotal to the decision to proceed to large-scale phase III testing or to abandon the compound after phase II testing" (DiMasi et al., 2015). ↩