Measuring Patient Similarity

Abstract

Given a query and a corpus of high-dimensional vectors, we consider the problem of recovering other items that are *similar* to the query. This is a challenging problem in machine learning not only because the notion of similarity is ill-defined in high-dimensional data but also because successful recovery depends on finding and encoding contextual cues from the query into the notion of similarity used to compare items in the corpus. Our problem is motivated by the need to have accurate ways for clinicians in hospitals to find similar patients from Electronic Health Record (EHR) data. We begin our investigation looking at how simple techniques like logistic regression and k-Nearest Neighbors perform in such a task.

1 Introduction

Measuring similarity between patients is a challenging task. With the advent of large Electronic Health Record (EHR) data becoming increasingly available, there is a need to develop measures of how patients differ from one another. Such measures could be used in downstream clinical applications either to help doctors find patients of interest, to match patients for clinical trials or even to find viable candidates for organ donation.

However, the problem is not without challenges. Medical data is temporal, heterogenous, high-dimensional and often has many missing features. Developing measures of similarity for such data is an active area of research.

In this report, we will detail our procedure and progress to building patient similarity metrics. We begin by providing some background on the methods we will use followed by a more detailed exploration of the task that we consider - selecting patients for randomized control trials. We then consider simple methods like nearest neighbor classifiers as well as more complex methods such as using representations from learned deep generative models and variational autoencoders (Rezende *et al.* , 2014; Kingma & Welling, 2014) and study how such methods may be applied to measure similarity between patients.

Related Work: There are two broad classes of methods - supervised and unsupervised. Supervised similarity metrics assume that a doctor hand-labels a small subset of patients as being similar (and potentially dissimilar too). The methods often rely on building parametric models that realize a similarity between elements in the set. (Zhu *et al.* , 2016; Wang & Sun, 2015; Sun *et al.* , 2012) all rely on tuning different kinds of linear and non-linear metrics between patients based on a small subset of clinical labels. Unsupervised methods assume nothing about the query - but instead rely on having a representation wherein *similar* patients are close to one another. One such example is that of Huang *et al.* (2014), who train Latent Dirichlet Allocation (Blei *et al.* , 2003) on patient records from EHR data and use the learned topics for a patient as representations under which similarity is measured.

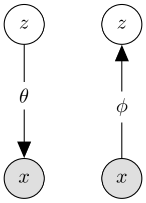


Figure 1: **Deep generative model:** The mode comprises a single latent variable z with the conditional probability $p(x|z)$ defined by a deep neural network with parameter θ. On the right, $q_\phi(z|x)$, the inference network, parameterized by ϕ, is used for inference.

2 Background

Generative Model: We consider learning in generative models of the form shown in Figure 1. We observe a set of D binary vectors $x_{1:D}$, where x_{dv} denotes whether the diagnosis code at index $v \in \{1, \ldots, V\}$ appears in patient d. We assume x_d was generated via the following generative process:

$$\begin{aligned} &z_d \sim \mathcal{N}(0, I);\ \gamma(z_d) \equiv \sigma(\text{MLP}(z_d;\theta)); \qquad (1)\\ &x_{dv} \sim \text{Bernoulli}(\gamma(z_d)) \end{aligned}$$

That is, we draw a Gaussian random vector, pass it through a multilayer perceptron (MLP) with parameters θ, pass the resulting vector through the sigmoid (denoted σ) sample from the resulting product of independent Bernoulli distributions (one per feature).

Variational Learning: We need to approximate the intractable posterior distribution $p(z|x)$ during learning. (For ease of exposition we drop the subscript on x_d when referring to a single data point.) Using the well-known variational principle, we can obtain the lower bound on the log marginal likelihood of the data (or $\mathcal{L}(x;\theta,\phi)$) in Eq. 2. where the inequality is by Jensen's inequality.

$$\log p(x;\theta) \geq \underbrace{\mathop{\mathbb{E}}_{q_\phi(z|x)}[\log p_\theta(x|z))] - \text{KL}(\,q_\phi(z|x)||p(z)\,)}_{\mathcal{L}(x;\theta,\phi)}, \quad (2)$$

We leverage an *inference network* or *recognition network* (Hinton *et al.* , 1995), a neural network which approximates the intractable posterior, during learning. This is a parametric conditional distribution that is optimized to perform inference. Kingma & Welling (2014); Rezende *et al.* (2014) use a neural net (with parameters ϕ) to parameterize $q_\phi(z|x)$. The challenge in the resulting optimization problem is that the lower bound (2) includes an expectation w.r.t. $q_\phi(z|x)$, which implicitly depends on the network parameters ϕ. This difficulty is overcome by using *stochastic backpropagation*.

With a normal distribution as variational approximation we have that $q_\phi(z|x) \sim \mathcal{N}(\mu_\phi(x), \Sigma_\phi(x))$. $\mu_\phi(x), \Sigma_\phi(x)$ are functions of the observation x. A simple transformation allows one to obtain unbiased Monte Carlo estimates of the gradients of $\mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z))]$ with respect to ϕ. If we assume the prior $p(z)$ is also normally distributed, the KL and its gradients may be obtained analytically.

3 Methodology

Feature extraction: The dataset we use for the experiments discussed in this document is composed of features extracted from the MIMIC database. More sepcifically rows the dataset correspond to individual admissions and columns are features. The rows are all represented as one-hot-encodings where an index takes the value 1 if the feature at that index was recorded for the admission. The total set of features were taken from unique prescriptions, procedures and diagnoses events from the MIMIC. We did not actually use the codes verbatim, but found UMLS concepts that most closely matched each code. Unified Medical Language System (UMLS) concepts are a hierarchical categorization of medical concepts which seved to combine icd-9 codes associated with similar medical concepts, and to put drug codes and ICD-9 codes in the same space. Since the number of features at this point was $\approx$ 20,000, we removed the rarest features to trim it down to ≈ 800, with $\approx$ 50,000 admissions in total. We then split the dataset into train, validation and test subsets which have sizes observing the numbers in 1.

Table 1: Train-Test-Validation

	Mortality	Aspirin	Cancer
Examples in train	41282	2037	734
Examples in validation	5899	623	231
Examples in test	11795	304	111

Sizes of the train/test/validation splits for different labels.

4 Cohort Creation

In this section we describe the process of creating cohorts from the MIMIC iii (MIMIC) dataset. For the purposes of our tasks a cohort will be a group of admissions that we know are similar in a clinically important way. The motivation for getting this sort of similarity signal is to use it for both the supervision and evaluation of our models and representations. One good source of descriptions of cohorts are randomized control trials (RCTs) and the inclusion/exclusion criteria that define them. A good repository of clinical trials information is clinicaltrials.gov, where thousands of RCTs and their inclusion/exclusion criteria are documented and indexed. We first want to further refine the large pool of RCTs before beginning to evaluate them more closely, we do so based on the following desiderata:

1. The RCTs should be related to the ICU setting, which is the context in which the MIMIC dataset was compiled.

2. The RCTs should have inclusion/exclusion criteria with clear analogues in the MIMIC data that we can query for.

The first stipulation was satisfied by looking at RCTs that returned from the query *ICU or Intensive Care Unit*. This shrunk the number of RCTs to around 1000, which was a bit more manageable to examine. We then proceed to evaluate the RCTs on the merits of the inclusion/exclusion criteria.

4.1 Looking at Randomized Control Trials

To simplify the examination process we first collected XML versions of the documents and assembled just the inclusion/exclusion criteria into a list for easy inspection. To give us some intuition regarding the natural grouping of the RCTs, we clustered the raw inclusion/exclusion criteria. This was done using simple k-means after binning the counts, doing one-hot-encoding and applying a TFIDF transformation. This gave a little more structure in the evaluation process and it allowed us to notice trends in the variability and types of criteria. We manually went through any RCT that did not obviously disqualify itself from consideration and checked to see whether they could be reconstructed from MIMIC admissions. We also payed attention to whether the trials were clinically relevant and interesting, such that our eventual experiments would be as well.

This filtering stage resulted in 5 candidate cohorts which could only be further evaluated after attempted reconstruction from MIMIC data.

1. *Mortality in Cancer Patients Admitted to the Intensive Care Unit in a Resource-limited Setting*
2. *Tracheostomy and Weaning From Mechanical Ventilation: Evaluation of the Lung Ultrasound Score*
3. *A Retrospective Review of a Comprehensive Cohort of Septic Shock: Assessment of Critical Determinants of Outcome*
4. *Dexmedetomidine for Sepsis in ICU Randomized Evaluation Trial (DESIRE)*
5. *Aspirin for Treatment of Severe Sepsis*

The next task was to find admissions in MIMIC that belong to the cohorts delineated by the criteria from the RCTs above. We did this by taking every criteria and finding an ICD-9 code to represent it. Since its possible to search for admissions for which a certain ICD-9 code was recorded, if we translate the RCT into a set of inclusion ICD-9 codes and a set of exclusion ones, we can easily find admissions corresponding to it. After doing this for every individual criteria, we assembled the cohorts themselves by taking the intersection of IDs in the inclusion criteria minus the intersection of IDs in the exclusion criteria. Practically speaking, we now were left with a list of admissions belonging to each of our cohorts.

At this stage we have three additional ways to evaluate the cohorts:

1. Look at how many matches in MIMIC were found for a given criteria.
2. Look at the size of the cohorts once we take the appropriate intersections and unions of criteria.
3. Look at the size of the intersections of the cohorts to see if they are reasonably sized.

These guidelines eventually led to the elimination of all but two cohorts. More specifically, the other potential cohorts had either too many admissions in the cohort, too few matches in MIMIC or too simple inclusion/exclusion criteria to begin with.

The final two cohorts come from the RCTs *Mortality in Cancer Patients Admitted to the Intensive Care Unit in a Resource-limited Setting*, and *Aspirin for Treatment of Severe Sepsis*, and we will refer to them as Cohort A and Cohort B respectively.

Cohort A is composed of 1076 patients and Cohort B is composed of 2964 patients. Their intersection is composed of 9 admissions.

4.2 Approximations

To create the cohorts a few approximations had to be made. We will discuss some of these approximations here and their implications.

Beginning with Cohort A: the following inclusion/exclusion criteria all had no clear analogous code that we could use to query MIMIC, so we chose to proceed without them after determining that the omission's impact on the integrity of the cohort was minimal.

- **Inclusion criteria:** *Functional classification state between 0-3*
- **Exclusion criteria:** *Absence of cancer recurrences*
- **Exclusion Criteria:** *Functional classification state of 4*

For cohort B there were two reasons we chose proceed without including a criteria. The first is the same as above, where the criteria had no clear analogous code. The second was when the code just did not match any patients in the mimic database. For both cases we only dropped the criteria after determining that the omission's impact on the integrity of the cohort was minimal.

Starting with the criteria that had no clear analogue or search-able criteria:

- **Exclusion criteria:** *Perspective of death in less than 24 hours*
- **Exclusion criteria:** *Restrictions on investment by the attending physician*

- **Exclusion criteria:** *diseases requiring the previous use of ASA*
- **Exclusion criteria:** *Use of another investigational medication in the last 30 days*

Now we look at the criteria that did have a search-able code or attribute but did not return any matches when queried for in the MIMIC database. Since they mostly consist of exclusion criteria, the fact that they are missing from mimic does not seriously compromise the integrity of the cohort.

- **Exclusion Criteria** *Pregnancy*
- **Exclusion Criteria** *analgesic allergy*
- **Exclusion Criteria** *precence of active peptic ulcer disease*
- **Exclusion Criteria** *precence of an epidural catheter*

At the end of this procedure we were left with two additional sets of labels for our dataset, each of which provide supervision and a basis for evaluation for the models and representations described in the rest of this document.

5 Evaluation

5.1 Precision

A commonly used metric in information retrieval is *precision at top k*, where we take a test query, find the k nearest points using some distance and check the fraction of those points that are of the same label as our query. We then average this value over the number of points in our query set.

To evaluate the Cancer and Aspirin cohorts, we caculate the precision at K using the cosine similarity as our distance and varying the source of the points. Our query points will always come from the test set, but we vary whether the nearest neighbors come from a test or train set. In this context this can be seen as checking for similar patients in the full history of patients versus finding similar patients that are in the hospital at the same time period.

In figure 4 we see the resulting curves when we use the aspirin cohort labels as supervision and in 5 we see the the curves for when we use the cancer labels as supervision. For both figures, the plots for train and test are shown. We also show the curves for when our query and neighbor points come from the raw feature space or from the latent representation. As a baseline we also show with a dashed line the expected precision if the neighbor points were taken at random.

5.2 MIMIC III (Johnson *et al.* , 2016)

This dataset comprises patient admissions into the ICU. We consider a task of recovering patients that satisfy the inclusion criteria for randomized control trials (RCT). The first evaluated mortality of cancer patients admitted to an ICU ward and the second RCT considered the efficacy of aspirin in the treatment of sepsis.

We first learn two deep generative models on the binary MIMIC dataset. One deep generative model is augmented to also perform classification of patient mortality labels during learning. We denote this with the suffix “ssi”.

We want to compare how well we can classify patients into RCT cohorts using different representations and different methods of classification.

On the x-axis, we vary the number of positive training examples (where the label corresponds to inclusion in an RCT). We train the following classifiers for each point on the x-axis.

1. Nearest neighbors (NN) in the input representation: denoted NN-x
2. NN in the posterior distribution under a model: denoted NN-mu
3. Logistic Regression (LR) trained on the posterior means under a model: denoted LR-mu

We then evaluate our classifiers on 100 positive and 100 negative patients obtained from the training and held-out set. We compute and plot the area under the receiver operator curve on the y-axis in Figure 2.

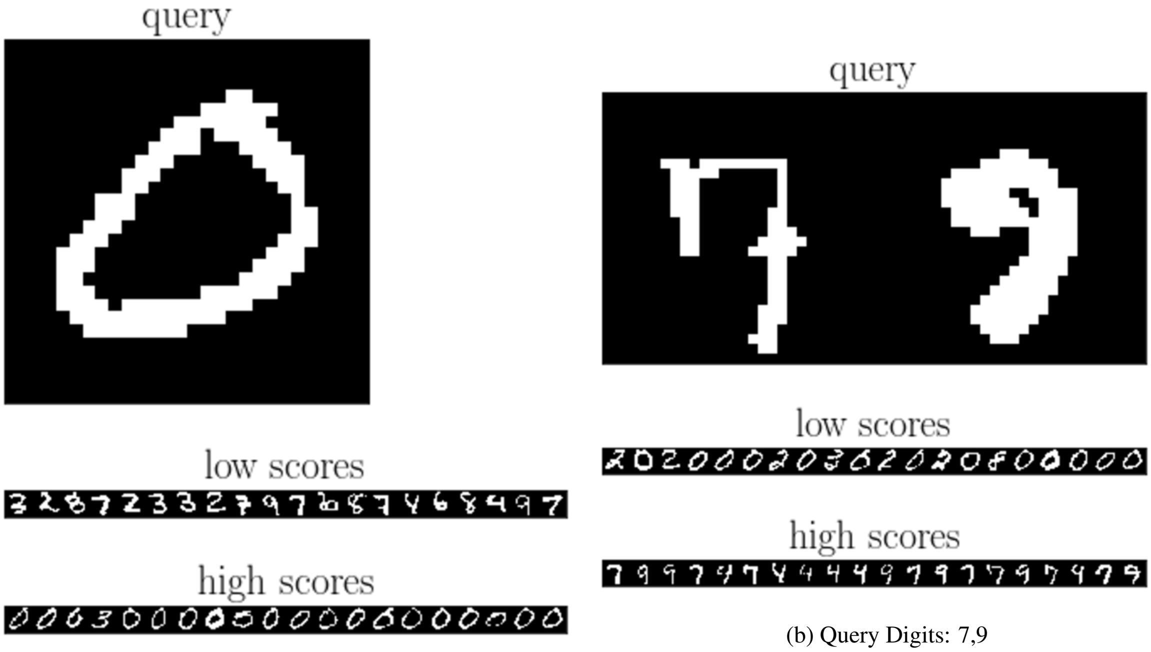
Overall, we find that despite the class imbalance for small numbers of positive patients in the training set, logistic regression trained on the posterior means outperforms other doing direct nearest classification on the latent space directly. This makes sense since the latent space is not directly optimized to perform well for a particular choice of query.

5.3 Binarized MNIST

The above analysis assumes that the query comprises a single patient. However, one can imagine scenarios where the doctor would like to find similar patients to a handful of people. How can one formalize this notion?

While we omit details here for brevity, but we provide preliminary experiments detailing the proof of concept for a technique we develop called *Latent Variable Sets*. The underlying intuition is that one can build to build a metric from a generative model by marginalizing out the latent variable. The procedure allows us to build query using multiple elements.

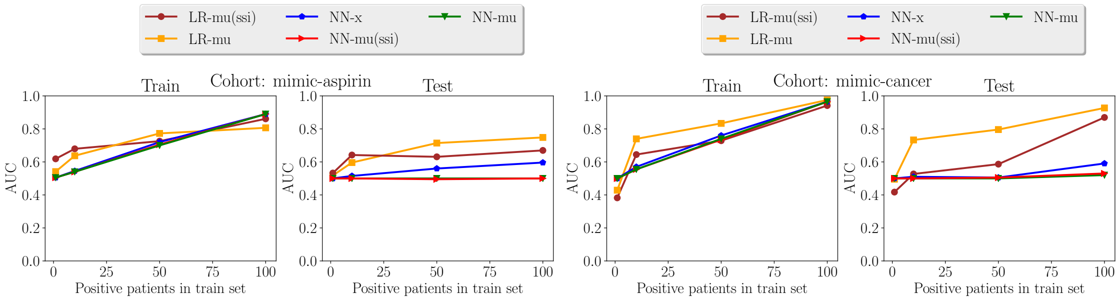
We use a model trained on the *MNIST* (LeCun *et al.* , 1998) dataset as proof of concept. We perform no preprocessing other than binarizing pixel values. We consider a single digit as a query and use the generative model to rank other digits. Under this metric visualize digits with both low and high scores. While not entirely representative of the patient scenario, we see that the algorithm produces rankings that are consistent with the query in Figure **??**. When the digit is zero - the other highly ranked digits are also zero. When the query is seven and nine - the other digits comprise other sevens and nines. While this is a toy problem, it does indicate that the method holds promise as a direction for future exploration.



(b) Query Digits: 7,9

(a) Query Digit: 0

Figure 3: **Latent Variable Sets:** Latent variable sets on binarized MNIST digits.



(a) Aspirin Cohort

(b) Cancer Cohort

Figure 2: **Predicting cohort inclusion**

References

Blei, David M, Ng, Andrew Y, & Jordan, Michael I. 2003. Latent dirichlet allocation. *JMLR*.

Hinton, Geoffrey E, Dayan, Peter, Frey, Brendan J, & Neal, Radford M. 1995. The" wake-sleep" algorithm for unsupervised neural networks. *Science*.

Huang, Zhengxing, Dong, Wei, Duan, Huilong, & Li, Haomin. 2014. Similarity measure between patient traces for clinical pathway analysis: problem, method, and applications. *IEEE Journal of Biomedical and Health Informatics*.

Johnson, Alistair EW, Pollard, Tom J, Shen, Lu, Lehman, Li-wei H, Feng, Mengling, Ghassemi, Mohammad, Moody, Benjamin, Szolovits, Peter, Celi, Leo Anthony, & Mark, Roger G. 2016. MIMIC-III, a freely accessible critical care database. *Scientific data*, **3**.

Kingma, Diederik P, & Welling, Max. 2014. Auto-encoding variational bayes. *In: ICLR*.

LeCun, Yann, Cortes, Corinna, & Burges, Christopher JC. 1998. *The MNIST database of handwritten digits*.

Rezende, Danilo Jimenez, Mohamed, Shakir, & Wierstra, Daan. 2014. Stochastic backpropagation and approximate inference in deep generative models. *In: ICML*.

Sun, Jimeng, Wang, Fei, Hu, Jianying, & Edabollahi, Shahram. 2012. Supervised patient similarity measure of heterogeneous patient records. *ACM SIGKDD Explorations Newsletter*.

Wang, Fei, & Sun, Jimeng. 2015. PSF: a unified patient similarity evaluation framework through metric learning with weak supervision. *IEEE journal of biomedical and health informatics*.

Zhu, Zihao, Yin, Changchang, Qian, Buyue, Cheng, Yu, Wei, Jishang, & Wang, Fei. 2016. Measuring Patient Similarities via a Deep Architecture with Medical Concept Embedding. *In: Data Mining (ICDM), 2016 IEEE 16th International Conference on*. IEEE.

A Appendix A: Baselines

We present here the results of a battery of baseline experiments performed to evaluate the models, task and cohorts.

We first train a set of simple classifier on the data, these classifiers include logistic regression, random forest and k nearest neighbors. We perform K-Fold validation with k=3 to find suitable hyperparameters, and report train and test AUC. We also comment on the hyperparameters chosen by the classifier. We find these baselines for two sets of input features and 3 types of supervision labels. The input features are either the raw one hot encodings from mimic or the means inferred by a variational autoencoder after learning with the raw features. The supervision labels are either mortality, the aspirin cohort or the cancer cohort.

Table 2: Evaluating *aspirin* in latent space

Classifier	AUC in train	AUC in test
Logistic Regression	.88	.87
Random Forest	.96	.512
KNN	.59	.56

Classification using the asspirin cohort labels as supervision and latent representation as input.

Table 3: Evaluating *cancer* in latent space

Classifier	AUC in train	AUC in test
Logistic Regression	.92	.92
Random Forest	.99	.57
KNN	.67	.63

Classification using the cancer cohort labels as supervision and a latent representation as input.

Table 4: Evaluating *mortality* in latent space

Classifier	AUC in train	AUC in test
Logistic Regression	.87	.87
Random Forest	.99	.7
KNN	.75	.71

Classification using the mortality labels as supervision and a latent representation as input.

Table 5: Latent space hyperparameters

Classifier	Mortality	Aspirin	Cancer
Logistic Regression (Regularization, C)	L1, 10	L1, 100	L1, 1000
Random Forest (N Estimators, criterion)	4000, Gini	100, entropy	3000, entropy
KNN (n neighbors)	6	6	6

The hyperparameters chosen for the classifiers through k-fold cross validation when the input is a latent representation.

Table 6: Evaluating *aspirin* in feature space

Classifier	AUC in train	AUC in test
Logistic Regression	.98	.98
Random Forest	.97	.84
KNN	.63	.57

Classification using the aspirin cohort labels as supervision when the raw features as input.

Table 7: evaluating *cancer* in feature space

Classifier	AUC in train	AUC in test
Logistic Regression	.97	.98
Random Forest	.99	.68
KNN	.7	.65

Classification using the cancer cohort labels as supervision and the raw features as input.

Table 8: Evaluating *mortality* in feature space

Classifier	AUC in train	AUC in test
Logistic Regression	.8	.79
Random Forest	.99	.81
KNN	.74	.70

Classification using the mortality labels as supervision and raw data as input.

Table 9: Feature space hyperparameter notes

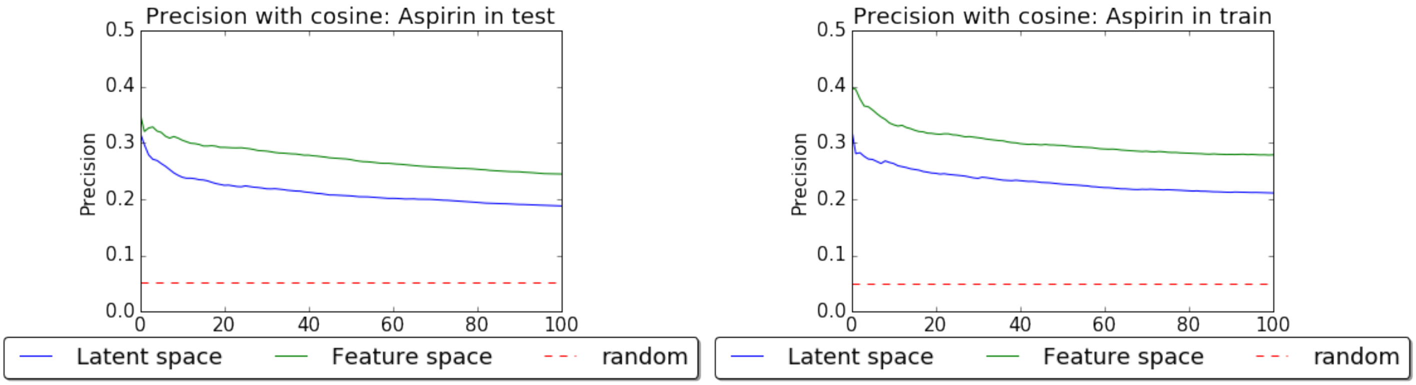
Classifier	Mortality	Aspirin	Cancer
Logistic Regression (Regularization, C)	L1, 10	L1, 100	L1, 1000
Random Forest (N Estimators, criterion)	4000, Gini	100, entropy	3000, entropy
KNN (n neighbors)	6	6	6

The hyperparameters chosen for the classifiers using k-fold cross validation when the raw features are used as input.

Table 10: Clinical Trial names and searchable codes

Cohort Name	**clinicaltrials.gov code**
Mortality in Cancer Patients Admitted to the Intensive Care Unit in a Resource-limited Setting	NCT02659839
Early Mobilisation in Intensive Care Unit ...	NCT02872792
tracheostomy and Weaning From Mechanical Ventilation...	NCT01793363
Aspirin for Treatment of Severe Sepsis	NCT01784159
A Retrospective Review of a Comprehensive Cohort of Septic Shock...	NCT01775956
Dexmedetomidine for Sepsis in ICU Randomized Evaluation Trial (DESIRE)	NCT01760967

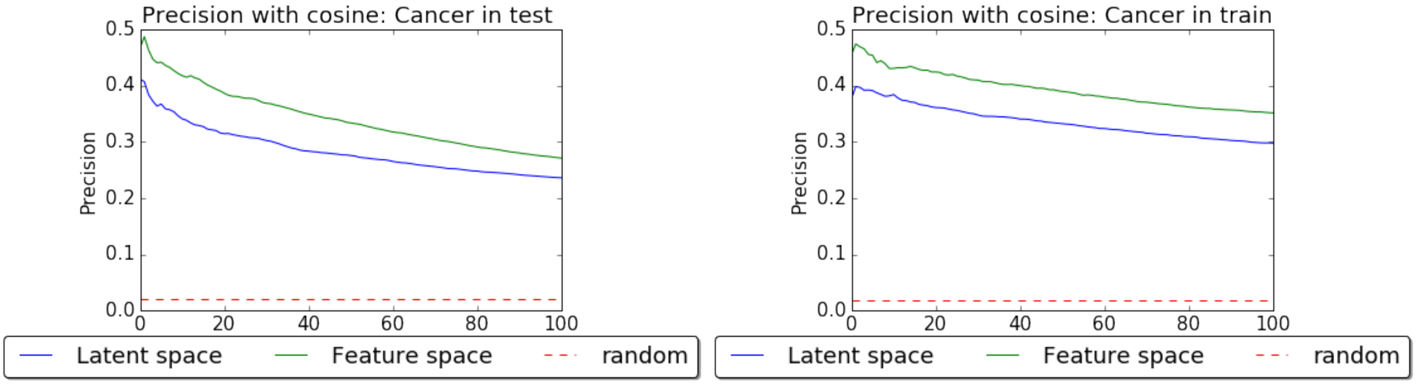
Every RCT title discussed with the code it is indexed by on the clinicaltrials.gov



(a) Aspirin test

(b) Aspirin train

Figure 4: **Precision: Aspirin cohort**



(a) Cancer test

(b) Cancer train

Figure 5: **Precision: Cancer cohort**