A Machine Learning Tutorial for Veterinarians: Examples Using Canine Atopic Dermatitis

Nathan Bollig[1,2]*, Douglas DeBoer[3], Dörte Döpfer[3]

This tutorial introduces machine learning (ML) topics using veterinary dermatology data from a US veterinary teaching hospital. We train binary classification models to predict treatment success of patients with canine atopic dermatitis and to predict a patient's status as a case or control. Important concepts relevant to the analysis are discussed in this article, and the code is made available in a Jupyter notebook format. Our results suggest that prediction models are not successful in performing these tasks. The analysis highlights key steps involved in training ML models and concludes with important takeaways for non-data scientists in several types of data management roles.

Introduction to Machine Learning

Machine learning (ML) is a branch of artificial intelligence focused on methods for developing computer programs that can learn automatically with experience [1]. In the machine learning paradigm, instead of incorporating direct instructions for how to carry out a task, a computer program is given information from which it learns to carry out a task.

Put differently, data represents the experience from which a ML system learns. Most generally, one can think of data as a set of examples of something, e.g. a collection of medical records, DNA sequences, movie reviews, or images. An example is sometimes referred to as an **instance** (e.g. one medical record, image, etc.). Each instance is represented as a collection of attributes or **features**, e.g. a medical record may be represented by a set of laboratory test results, or an image may be represented by a grid of pixel values. The **feature representation** of the data describes how objects in the real world will be represented as input to the ML system.

An instance may also have an associated **class label**. For example, a medical record may have a **binary** label (i.e. has one of two possible values) indicating whether the patient was diagnosed with Addison's Disease, or an image of a house could have a label indicating its current market value. Although a label is an attribute of a data instance just like a feature, it is helpful to differentiate features (inputs to a predictive model) from labels as target attributes that we want the system to be able to predict.

In the **supervised** machine learning approach, the goal is to design a system that translates some input (features of a data instance) into some output (its label). For example, a ML model may be given the laboratory test results in a patient's medical record and asked to predict whether the patient has Addison's Disease. Or a ML model may be given a digital image of a house and asked to predict its current market value.

What does it mean for a machine to learn? With respect to the specific task that the machine is to perform, **learning** means that it becomes more accurate as it accumulates more experience. Equivalently, as the system is given more examples of inputs and the appropriate labels, it becomes more accurate in its ability to predict the correct labels for new instances that it has not seen before. Often a model is tested by splitting labeled data into a **train** and **test** pool. The model is created (**trained**) using the data in the train pool, it is used to make predictions on the instances in the test pool, and then the

[1] Department of Computer Sciences, University of Wisconsin-Madison, Madison, WI
[2] Department of Pathobiological Sciences, School of Veterinary Medicine, University of Wisconsin-Madison, Madison, WI
[3] Department of Medical Sciences, School of Veterinary Medicine, University of Wisconsin-Madison, Madison, WI
* nbollig@wisc.edu

model’s accuracy is evaluated by comparing the predicted test labels to ground-truth (actual) labels. **Performance scores** or **metrics** other than raw accuracy are common, including **balanced accuracy**, **F1-score**, and **area under receiver-operating characteristic** or **precision-recall curves** (AUC measures).

There are many types of **learning algorithms**, all representing different ways to learn a mathematical function that maps input features to output labels. Common machine learning algorithms, such as **random forests**, **nearest neighbors**, **support vector machines**, **naïve Bayes**, **regression**, and **neural networks**, are simply different approaches for learning such a mathematical function.

This article and accompanying presentation represents an attempt to bring these fundamental principles to life in the context of a simple veterinary data science problem. For a non-technical introduction to ML, references [2] or [3] are recommended. For a deeper introduction, see [1] or [4]. Widely-used terms throughout the ML and data science communities are formatted in bold throughout this paper.

Introduction to Canine Atopic Dermatitis

Canine atopic dermatitis (CAD) is a common inflammatory skin disease often associated with IgE antibodies targeting environmental allergens [5], and one standard treatment for over 50 years has been allergen specific immunotherapy (ASIT) [6]. In humans and dogs, ASIT is the only type of allergy treatment that can potentially reverse the progression of disease, has long-term efficacy, and is not associated with significant short- or long-term adverse effects [7]. Observational studies report response rates to ASIT (usually measured as at least a 50% improvement in clinical signs) ranging between 50% and 100% [6,8]. Few controlled studies exist, but one placebo-controlled trial of 51 dogs reported a 84% chance of greater than 50% improvement in clinical signs [9].

Modern ASIT is administered as subcutaneous immunotherapy (SCIT) or as sublingual immunotherapy (SLIT). The latter has been a popular treatment of human atopic dermatitis for many years, with studies reporting conflicting measures of efficacy [6]. Based on human studies in Europe, a 2009 position paper published by the World Allergy Organization concluded that SLIT is effective and has a safety profile not worse than SCIT [10], however the use of SLIT in the United States has been limited in people and only recently reported in dogs [11,12]. The efficacy of SLIT needs to be evaluated in canine patients in the US, and further study is needed to characterize factors that could influence treatment outcomes.

Human studies suggest that atopic dermatitis is associated with a parental history [13,14], and that certain ethnicities are more susceptible [15,16], emphasizing that genetics play an important role in the development of this disease. Consistent with this hypothesis, studies on canine populations in Europe have reported breed predilections for canine atopic dermatitis, generally implicating West highland white terriers, Boxers, and other Bulldog breeds [17,18]. In the US, multiple studies have implicated Boxers, Terriers, and other breeds, but regional differences in breed predilection have not been explored. Most importantly, many reports do not compare CAD breed prevalences to the profile of breeds presenting to the veterinary clinic, making it difficult to confidently characterize specific at-risk breeds across locations [19].

Prior to 2001, there had been inconclusive results on the potential association between sex and atopic dermatitis. Studies published after 2001 agree that CAD does not display sex predilection generally [20], although sex was reported as significant within Boxers and Golden retrievers in one study [21].

Environmental factors associated with canine atopic dermatitis have been proposed. Mixed results exist regarding the influence of season of birth. A research group in Sweden found more CAD in dogs born in autumn months [22], but the same group also found a lack of significant association in a later study [18]. To the authors’ knowledge, no analysis evaluating the connection between season of birth and

CAD has been conducted on a population of dogs in the US. Additionally, the presence of animals in the home has been shown to reduce the risk of atopic dermatitis in people [23–25], and two canine studies indicated that cohabitation with other dogs or cats reduces CAD likelihood [17,26]. This is consistent with the "hygiene hypothesis", a well-characterized correlation in humans between living in an extremely clean environment and increased risk of allergic disease [20]. However, in the case of CAD, more canine studies are needed to separate the effect of environment type from potential region-specific or other confounding factors.

Defining a Toy Clinical Problem

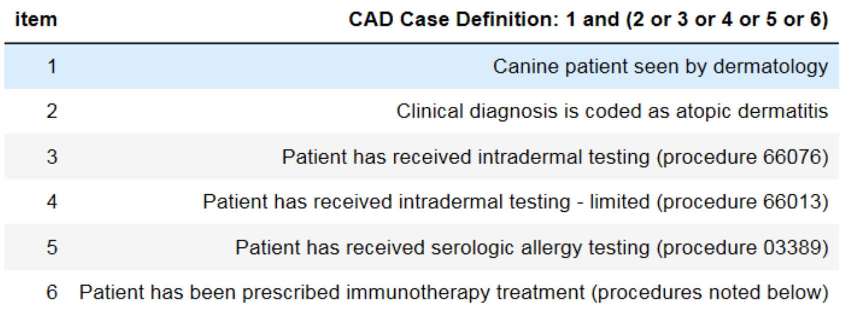
This tutorial will examine clinical data from a US veterinary teaching hospital with the goal of identifying factors predictive of CAD (1) treatment success and (2) case-control status using an ML modeling approach. A positive result would involve showing that some of the available features offer a predictive signal for one or both target variables, i.e. that the inputs can be used to predict the target(s). However, the primary goal of this tutorial is not to obtain and discuss a positive result, but to lay out an overview of how to develop and analyze ML models. Although we encounter mostly negative results, it is hoped that the surrounding discussion will nevertheless provide illuminating insights.

Assembling a Data Set

Clinical data was extracted from the electronic health record (EHR) at the University of Wisconsin Veterinary Medical Teaching Hospital (VMTH). Database queries were executed to generate electronic reports containing clinical data. These reports included nine lists of patients receiving specific medical procedures, one table of all patients, and a table of all visits. Using R statistical computing software [27] operating within the R Studio integrated development environment, these clinical reports were assembled into a data set of cases and controls. All non-canine species were filtered out through a combination of programming and manual review.

Cases were defined based on having received intradermal testing or immunotherapy procedures in the hospital (Figure 1).

Figure 1. Canine atopic dermatitis (CAD) case definition.



item	CAD Case Definition: 1 and (2 or 3 or 4 or 5 or 6)
1	Canine patient seen by dermatology
2	Clinical diagnosis is coded as atopic dermatitis
3	Patient has received intradermal testing (procedure 66076)
4	Patient has received intradermal testing - limited (procedure 66013)
5	Patient has received serologic allergy testing (procedure 03389)
6	Patient has been prescribed immunotherapy treatment (procedures noted below)

Patients treated with allergy shots were identified based on having received an initial allergy shot set. Treatment success was then defined using two variables. First, a binary classifier was generated, such that its value is positive (indicating "treatment success") if and only if a patient received a refill set. Second, as a potential numerical indicator of treatment success, the number of immunotherapy refills was calculated. Patients receiving sublingual immunotherapy were identified based on having received an initial sublingual treatment set, and binary and numeric treatment success variables were calculated in the same way, based on refills of sublingual treatment.

Controls were defined as a sample of canine dermatology patients not included in the case group.

The data set included the following data on each case and control patient: medical record number, name, breed, sex, ZIP code, date of birth, type of therapy received, date of initial treatment, age of patient at date of initial treatment, and DVM provider code at initial treatment. Initially, many control patients had missing data for breed, sex, date of birth, and ZIP code. Dermatology exam procedure queries were used to recover sex data for these control patients, and a table of the entire VMTH population was used to recover breed, date of birth, and ZIP code. Controls without dermatology exam codes or recoverable data from the full VMTH population table were dropped. Initially, 37% of patient dates of birth were entered with a century of “00” instead of “20”, and these were corrected by manual review instead of deleting them from the data set. Two test patients were identified and deleted.

Several variables were computed from the available data. Breeds were manually classified into the following categories: Spaniel, Retriever, Shepherd, Pointer, Hound, Bulldog breed, Terrier, Setter, Northern breed, Poodle, Toy breed, Pinscher, Large breed, Spitz, Mixed breed, Other. ZIP codes were assigned rural-urban continuum codes (RUCC), which characterizes county population numerically from 1 (largest) to 9 (smallest). Ages were categorized as “young” (less than 660 days) and “old” (at least 660 days) based on subjective partitioning of the age distribution. Season of initial treatment and date of birth were defined as winter (December, January, February), spring (March, April, May), summer (June, July, August), or fall (September, October, November).

Code Availability

All code for this tutorial is available in a Jupyter notebook format at: https://github.com/nathanbollig/ML-for-veterinarians.

Data Preprocessing

After assembling the raw data set, additional processing tasks are often required to make the data amenable to machine learning. Important data preparation steps include encoding categorial features, formatting, and imputation of missing values. Encoding categorial features refers to replacing strings with a numerical or vector encoding, such as replacing “male”, “female”, “spayed”, and “neutered” with the numbers 1, 2, 3, and 4, respectively (this technique is called **label encoding**). Formatting steps might involve things like taking only the first part of a zip code or converting dates to a common format or representation. Imputation refers to the process of replacing blank values of a feature with a guess, such as the mean of all remaining values. After processing, our dataset had columns as shown in Table 1.

Table 1. Available features in the processed data set.

breed_cat	Patient breed (Spaniel, Retriever, Shepherd, Pointer, Hound, Bulldog breed, Terrier, Setter, Northern breed, Poodle, Toy breed, Pinscher, Large breed, Spitz, Mixed breed, Other)
sex	Patient sex (female, male, neutered, spayed)
zip	ZIP code for patient address

RUCC	Rural-urban continuum codes (RUCC) characterizes county population numerically from 1 (largest) to 9 (smallest)
case	Case (1) or control (0)
dob	POSIX timestamp of patient date of birth
therapy	Patient therapy (allergy shot, sublingual, or none)
first_proc_date	POSIX timestamp of patient's first treatment date
first_proc_season	Season of patient's first season
age_days	Patient age at day of first treatment
age_cat	Ages are categorized as 1 ("young", less than 660 days) and 2 ("old", at least 660 days)
first_dvm_code	Numerical code representing attending DVM at first treatment
tx_success	Treatment success (1) or failure (0), where success is defined by patient returns > 0
returns	Number of return visits after initial treatment
dob_season	Season of patient's date of birth

Modeling a Prediction Task

Identifying outcome variable and candidate features

In the first part of this tutorial, we will create machine learning models to predict the `tx_success` outcome for a patient using the other variables as input features. In particular, we will ensure that the input representation of each patient does not include the `tx_success` variable, and since the `tx_success` is defined by the number of returns we need to ensure that the `returns` variable is also excluded from the list of input features

There are 411 cases in our data set for which treatment success data was available, and the treatment success rate was computed to be 74%.

We can visualize the data in the following way. The inputs can be organized into a large matrix *X* where each row represents a patient (a data instance) and each column represents a feature of that patient. There are 411 rows because there are 411 patients. There are 13 columns because there are 13 available features. We also need to keep track of the outcomes, e.g. as a list of values of `tx_success` corresponding to the rows in *X*.

Univariate feature selection

A common technique in machine learning is **feature selection.** This refers to the process of removing features, which sometimes leads to a better-performing or more accurate model. It can also be easier to interpret a model that relies on fewer features. If the input data is organized into a matrix *X* as described above, feature selection amounts to the process of removing columns from this matrix.

One of the simplest methods for feature selection is to retain the features that are most strongly associated with the outcome variable. For classification problems, the **chi-squared statistic** and **mutual information** are commonly used to measure the strength of the association between a given feature and the outcome variable. These approaches are univariate because they evaluate each feature independently from each other feature. Note that if the outcome happens to be a nontrivial function of multiple features, then these statistics may be very small even for important features. Therefore, the results of univariate feature selection should be taken lightly, but we will use this approach for simplicity and because it is supported by additional analysis on this data set.

Table 2 shows the 5 highest features ranked by chi-squared and by mutual information.

Table 2. Highest ranked features based on univariate comparison with the `tx_success` outcome variable.

Statistic	Feature	Value
Chi-squared	first_proc_date	11830000.00
	dob	9718000.00
	zip	255.20
	age_days	26.02
	therapy	12.16
Mutual information	zip	0.06139
	first_proc_date	0.05053
	therapy	0.04262
	first_dvm_code	0.03187
	sex	0.02841

Based on this information, we will restrict ourselves to at most the following features when training our ML models: `first_proc_date`, `zip`, `dob`, `breed_cat`, and `therapy`. Our input data array *X* now contains 411 rows and 5 columns. However, before training a model we first should look for any strong correlations between the candidate input features, by measuring the **correlation coefficient** between pairs of columns. When this is done, there is unsurprisingly a strong correlation between dob and `first_proc_date`. We will remove the `dob` feature as well, leaving now only `first_proc_date`, `zip`, `breed_cat`, and `therapy` as the final features.

The above decisions reflect a subjective interpretation of crude statistics, and it would have been reasonable to remove fewer features or even to include all features for model training. In the foregoing analysis, the impact of feature selection was to significantly reduce the run time of experiments without otherwise changing the results significantly.

Evaluating Model Performance

As explained in the introduction, data is typically split into a training set and testing set. This serves the important purpose of guaranteeing that data used to test the model does not appear in the set of data used to train the model, an instance of a problem known as **information leakage.** When information leakage is present, the measured performance of the model can be artificially inflated.

Because data used for training cannot be used for testing, a difficult tradeoff can emerge between having enough training data and enough testing data when analyzing small datasets. When the training set is too small, the learning algorithm may not have enough information to learn a high-quality model that will perform well on novel input. On the other hand, when the test set is too small, the performance score estimated on this data set will be less reliable (i.e. may have higher variance). Another drawback of splitting data into a single training set and testing set is that the resulting performance on the test set can be highly sensitive to the way that data is allocated between these sets.

A common evaluation technique called **k-fold cross-validation** addresses both problems. In this approach, data is partitioned into *k* subsets (often *k*=5 or 10), and we iteratively reserve one subset for the test set while the remaining *k*-1 subsets are used for training. A total of *k* models is trained (each with a different train and test set and no information leakage), and for each of them we can evaluate its performance on the held-aside test set. At the end of the *k* iterations, we average together the *k* performance scores to get a composite score. This is a common, data-efficient approach that uses all data for testing and is not reliant on a single train/test split. Note that since multiple models are trained (each with a different training set), the performance score computed by cross-validation evaluates the learning algorithm rather than a specific learned model.

The types of performance scores commonly used to describe the performance of a binary classification model on its test set are summarized in Table 3. The simplest and most familiar metric is **accuracy**, which simply reflects the percentage of the test set on which the model's predictions are correct. Importantly, accuracy can be less informative when the target outcome appears with prevalence significantly different from 50%, a condition known as **class skew**. For example, in our dataset the `tx_success` variable is positive about 74%, reflecting a moderate level of class skew. A model that indiscriminately predicts a positive label for every input, regardless of the values of its features, could for example achieve an accuracy of 74%.

It is often better to take additional factors into account when describing the performance of a binary classifier in the presence of class skew. The classifier's **recall** (or **sensitivity**) describes the proportion of true positive cases that are correctly identified as positive. Its **precision** (or **positive predictive value**) describes the proportion of predicted positive cases that are actually positive. You can see that these two metrics provide complementary information about the classifier's performance: recall indicates how "willing" the classifier is to identify a patient as having a treatment success, and precision indicates how often this positive label is correct. The **F1-score** represents an average of recall and precision. It uses a harmonic rather than arithmetic mean because the result is generally more sensitive to small values of recall and precision than to large values, making the F1-score more conservative than an arithmetic mean of recall and precision.

Table 3. Performance metrics. TP – total true positives; TN – total true negatives; FP – total false positives; FN – total false negatives.

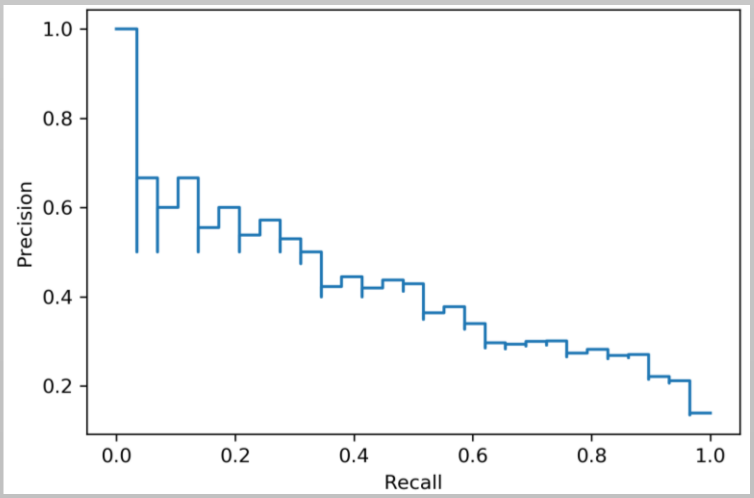
Statistic	**Definition**
Accuracy	(TP + TN) / total
Recall	TP / (TP + FN)
Precision	TP / (TP + FP)
F1 score	2 precision · recall / (precision + recall)

Note that the metrics in Table 3 are based on the notion of a binary classifier whose output is either positive or negative. However, many common ML models for binary classification output a single real number between 0 and 1 that can be calibrated to indicate the model's confidence in a positive prediction. A simple way to interpret this type of output is to treat a prediction as positive if and only if it exceeds the value 0.5. However, it is sometimes useful to consider lowering or raising this threshold to observe the effect on recall, precision, and therefore on the F1-score.

This is a central idea behind the **precision-recall (PR) curve**, which is explained in Box 1. The area under the PR curve (**PR AUC**) is a single number that reflects the precision and recall of the classifier across the full range of possible classification thresholds.

Box 1. Precision-recall curves.

Imagine setting the threshold for a positive prediction at exactly 1. This means that unless the model indicates 100% confidence in a positive prediction, no data instances will be labeled positive by the model. Most likely, the recall is zero and the precision is undefined. Now imagine lowering this threshold until eventually one data point has a prediction probability that exceeds it. This raises the recall from zero to a small number, and results in a precision of either 0 or 1 depending on whether the prediction was correct. As you imagine the threshold continuing to drop and more data points being labeled as positive, the recall continues to increase and precision may first stabilize to a relatively higher value (since the points first labeled as positive have a higher confidence and therefore may be more likely true positive). As the threshold drops below 0.5 and continues to fall, the recall increases further and precision drops as the system begins making many false positive predictions. Once the threshold falls below the smallest prediction probability, the recall is 100% and the precision is equal to the positive class prevalence. In this mental procedure of starting with the threshold at 1 and reducing it toward 0, you have traced out a precision-recall curve from left to right, an example of which is shown below.



Because treatment success occurred in 74% of patients, a classifier that indiscriminately labels patients as having a treatment success is correct 74% of the time on this dataset. This result suggests that the accuracy of a ML model would need to exceed 74% in order to conclude that it is better than a model that effectively does nothing. A priori, it is not obvious what level of F1-score or PR AUC a ML model should exceed in order to conclude that it is extracting a predictive signal. To reason about this, we need to compute the F1-score or PR AUC of a classifier that indiscriminately predicts positive, and compare our ML models' performance to that value. Such a trivial model used for comparison is called a **null** or **no-information** baseline classifier because it does not use any of the input features to make its prediction. Although there are multiple ways to define null models, the indiscriminately positive classifier is generally considered ideal in precision-recall analysis [28]. We will ignore the accuracy of this baseline classifier since this classifier is not an appropriate baseline for accuracy comparison.

In this tutorial, 95% confidence intervals are computed using **bootstrapping** as in [29,30].

Comparing Machine Learning Algorithms

No free lunch

As explained in the introduction, a machine learning model is nothing more than a mathematical function that maps input (e.g. the features that describe our patients) to some output variable (e.g. the treatment success). Popular **learning algorithms** such as random forests, support vector machines, and neural networks differ widely in how they encode and learn the underlying mathematical function that maps input to predictions.

Why are different types of learning algorithms necessary? Based on a central result of statistical learning theory, we know that there is no universal learning algorithm, i.e. no algorithm that can succeed on all possible learning tasks. More precisely, the theory states that for any learning algorithm there will be a task for which it will fail to learn a successful model, while a different learning algorithm would be successful [31]. An implication of this so-called **no-free-lunch principle** is that it is a good idea to empirically compare multiple learning algorithms to see which performs best for the task at hand.

Hyperparameter tuning

Some learning algorithms require certain parameters to be specified before starting to train a model. These are called **hyperparameters** to distinguish them from the parameters learned during model training. In some cases, the hyperparameter values can dramatically impact the quality of the learned model, and in other cases they may have a minimal impact. For each learning algorithm, it is typically known from empirical evidence which hyperparameters need to be carefully adjusted in order to optimize the performance of the model.

The process of selecting the value for a hyperparameter is called **hyperparameter tuning**. One approach to tuning a hyperparameter would be to run a 10-fold cross-validation experiment several times, each with a different value of the hyperparameter, and then pick the hyperparameter that results in the best performance score. If multiple hyperparameters need to be tuned, this exploration can be done over all possible combinations of a set of values for each hyperparameter, an approach known as **grid search** (explained further in Box 2). However, it is important to note that using test set performance scores for hyperparameter tuning represents an example of information leakage (since hyperparameters are part of the training input to the learning algorithm).

Ideally, we would utilize a separate **tuning** data set only for hyperparameter tuning and not for final testing of the model. However, in this case there is not enough data to dedicate to a separate tuning set, so we will use an approach called **internal cross-validation.** As explained in Box 2, internal cross-validation is a method for selecting the training hyperparameters for each of the *k* models in a *k*-fold cross-validation experiment without creating information leakage and without requiring a dedicated tuning set.

Experiment 1: Predicting treatment success

We performed 5-fold cross-validation to measure the performance of the following learning algorithms when used to predict CAD treatment success: support vector machine, random forest, decision tree, bagging trees, boosted trees, and logistic regression. The popular machine learning framework sci-kit learn [32] was used to perform these experiments. We used 5-fold internal cross-validation to perform grid search over a range of common values for each of the algorithm's hyperparameters. More details are available in the Jupyter notebook that accompanies this article.

Box 2. Internal cross-validation and grid search.

Recall that a cross-validation experiment involves iterating through a list of *k* folds, which we will refer to as “outer” folds to distinguish them from the folds established for inner cross-validation.

The outer cross validation experiment begins with specifying a partitioning of the data into *k* subsets, then we do the following.

```
For $i = 1, \dots, k$ do
    a. Take the $i$th subset of data as the test set.
    b. Take all remaining subsets of data as the training set.
    c. Before training a model on the training set, find the best hyperparameters
       using grid search.
    d. Train the model using the hyperparameters selected in Step c.
```

Step c (grid search) itself represents an iterative calculation. Since grid search is performed using cross-validation, the resulting process is called **internal cross validation**. Importantly, this grid search only uses the *i*th model’s training data established in Step b, so the process does not depend on the data that will be used to test the *i*th model.

Grid search involves enumerating a list of possible hyperparameter combinations for a given model type. For example, suppose that a random forest model is being trained, and we are considering values of hyperparameter 1 in the set {12, 15, 17}, hyperparameter 2 in {2, 3}, and hyperparameter 3 in {5, 7, 9}. There are 3x2x3 = 18 possible combinations. Each one is evaluated using (internal) cross-validation. The following summarizes what happens in Step c.

```
For each hyperparameter combination
    Specify a partitioning of the training set (for the $i$th model) into $n$ subsets.
    Assess performance using $n$-fold cross-validation.
```

In summary, this procedure involves three nested loops:

```
Loop over $k$ outer cross-validation folds
    Loop over hyperparameter combinations
        Loop over $n$ inner cross-validation folds
```

For each outer cross-validation fold, each candidate hyperparameter combination is tested *n* times, resulting in *n* performance scores that can be averaged into a composite score. The hyperparameter combination with the highest composite score is selected and then used for training the *i*th model.

The results of this experiment are shown in Table 4 below. We observe that all learning approaches perform at a comparable level, which is not surprising given that these are all known to be successful algorithms for learning classification models. The results of the cross-validation experiment appear promising (PR AUC around 88%), but remember that we need to compare the results against a null baseline as shown also in Table 4. Unfortunately, the baseline model achieves comparable performance to the machine learning models. Therefore, we conclude that none of the machine learning approaches can extract a predictive signal from the available features. In particular, there is no evidence from this data that immunotherapy treatment type is correlated with treatment success.

Table 4. Performance of machine learning models and a null baseline on the treatment success prediction task.

Model type	Accuracy	F1-score	PR AUC
Support vector machine	0.740 (0.698, 0.781)	0.851 (0.822, 0.879)	0.870 (0.849, 0.891)
Random forest	0.764 (0.723, 0.803)	0.862 (0.835, 0.888)	0.879 (0.857, 0.900)
Decision tree	0.747 (0.706, 0.788)	0.842 (0.809, 0.872)	0.880 (0.857, 0.903)
Bagging trees	0.757 (0.715, 0.798)	0.852 (0.822, 0.879)	0.880 (0.856, 0.902)
Boosting trees	0.736 (0.693, 0.779)	0.846 (0.817, 0.873)	0.868 (0.846, 0.889)
Logistic regression	0.740 (0.696, 0.784)	0.851 (0.822, 0.877)	0.870 (0.847, 0.892)
Null baseline	N/A	0.850 (0.822, 0.877)	0.870 (0.849, 0.891)

Experiment 2: Predicting CAD case vs. control status

In another experiment, we return to the original data and redefine the binary `case` variable as the outcome. In other words, we aim now to train a machine learning classifier to predict whether a given patient is a CAD case or control, based on the available features. As in our preparation for experiment 1, we similarly now remove all features relating to immunotherapy treatment, since those would immediately give away the patient's status as a CAD case. The remaining features are: `breed_cat`, `sex`, `zip`, `RUCC`, `dob`, and `dob_season`. In this setup, we have a case prevalence of 29.2% (657/2249).

Given that the feature set is reasonably small already, we did not perform feature selection for this analysis. We repeated the cross-validation experiment to assess the performance of the same learning algorithms in the CAD case vs. control prediction task, using the specified features. The results, along with the scores for a null baseline classifier, are presented in Table 5.

Table 5. Performance of machine learning models and a null baseline on the CAD case vs. control status prediction task. The support vector machine with linear kernel is used to discuss model transparency in the next section.

Model type	Accuracy	F1-score	PR AUC
Support vector machine	0.870 (0.856, 0.884)	0.816 (0.795, 0.836)	0.843 (0.828, 0.858)
Random forest	0.816 (0.800, 0.831)	0.662 (0.629, 0.692)	0.721 (0.693, 0.747)
Decision tree	0.803 (0.786, 0.819)	0.667 (0.638, 0.695)	0.715 (0.687, 0.740)
Bagging trees	0.813 (0.797, 0.829)	0.659 (0.628, 0.690)	0.718 (0.690, 0.745)
Boosting trees	0.881 (0.867, 0.895)	0.788 (0.761, 0.812)	0.826 (0.804, 0.847)
Logistic regression	0.708 (0.688, 0.727)	0.000 (0.000, 0.000)	0.646 (0.637, 0.656)
Support vector machine (linear)	0.459 (0.438, 0.479)	0.392 (0.365, 0.419)	0.504 (0.480, 0.529)
Null baseline	N/A	0.452 (0.431, 0.475)	0.646 (0.637, 0.655)

The results appear more promising because the ML approaches outperform the baseline classifier with statistical significance, based on the 95% confidence intervals indicated. We conclude that a predictive signal, with respect to predicting CAD case vs. control status, is present in the available features.

Model transparency

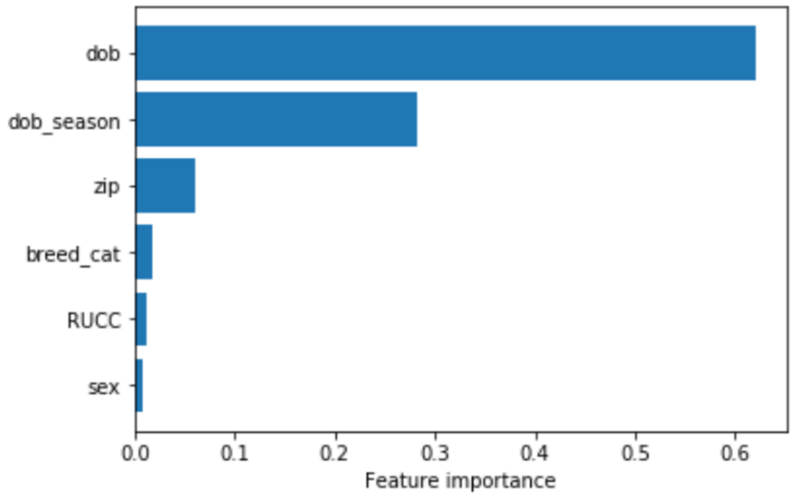
We will analyze the support vector machine, which is one of the most successful machine learning approaches in experiment 2. First, we look at the hyperparameters selected across the five external cross-validation folds (each one selected by an internal cross-validation loop), and select the best values of their 'cost' and 'gamma' hyperparameters, noting that all models utilized a **radial basis function (RBF)** rather than a linear kernel.

Even without describing what a support vector machine is, what these hyperparameters do, or the difference between a linear and RBF kernel, we can use this example to examine the issue of model **transparency**. For some ML models, like SVM with the linear kernel, it is easy to explain which features are most important for the model when making a prediction (i.e. to measure **feature importance**). It would, for example, be helpful to know that `breed_cat` or `sex` is considered important by the model, because then we might infer that there is some special relationship between the feature and the outcome. For this reason, SVM with a linear kernel learns a more transparent **white box model**, while SVM with an RBF kernel learns a **black box model**. While there are techniques for extracting feature importance information from black box models, this information can be difficult to obtain and less informative than feature importance derived from more transparent models.

Sometimes the choice of learning algorithm presents a tradeoff between transparency and performance. A linear support vector machine model is highly transparent but scores poorly on this task (Table 5). The RBF kernel allows the SVM to learn more complex, non-linear functions, which results in a less easily interpretable model that has higher performance.

The gradient boosted tree classifier is among the top classifiers in the CAD case vs. control prediction task, and the so-called reduction in **Gini impurity** during training can be used as a measure of feature importance. Following the same approach as above, we can manually examine the selected hyperparameters and infer a reasonable best combination of hyperparameters for training a single model over the entire dataset. We do this and report the resulting feature importance measures in Figure 2.

Figure 2. Feature importance for a gradient boosted tree model trained on the entire dataset, measured by the decrease in Gini impurity associated with each feature.



Hopefully the reader can appreciate the value of the gradient boosting tree granting easy access to feature importance information. A low-performing, transparent model is generally useless, since we cannot trust the feature importance information it provides. In this case, we are led to form the intriguing hypothesis that `dob` may have some relevant connection to the patient's CAD case vs. control status. We will explore this hypothesis in the next section.

Important Takeaways

Evaluation methodology is critical

Every machine learning model needs to be evaluated against a set of real, independent data. There should be no opportunity for the training set to present information to the model about the test set. The test statistic used should be appropriate for the situation (e.g. raw accuracy may not be the best metric). Finally, it is ideal to have confidence intervals or some other measure of variance of the reported values, although this practice is not universal in the machine learning literature.

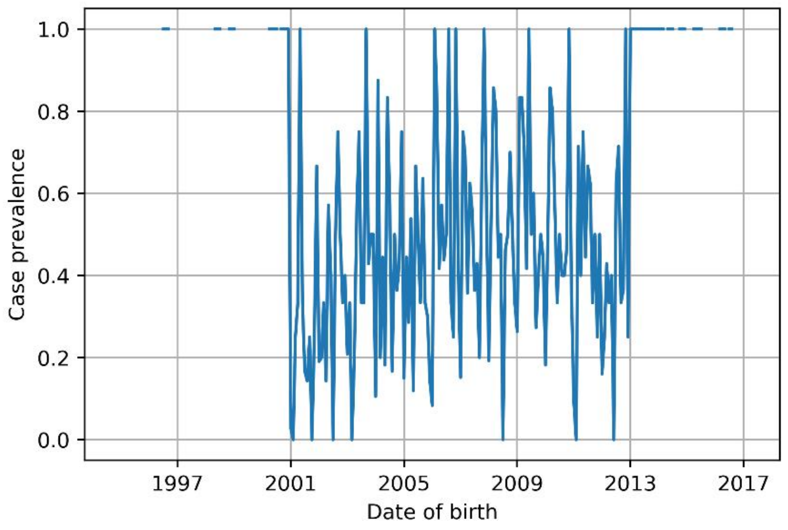
Feature importance does not designate a linear or causal relationship

We showed in the last section that the two most important features for the gradient boosted tree model were `dob` and `dob_season`. It is important to remember that strong importance does not designate a linear relationship between the feature and the log-odds of the case status, as it does in logistic regression. Therefore, for example, we do not conclude that case frequency increases for animals born at a later time, only that there is some relevant signal in the `dob` feature. Sometimes highly important features correlate negatively with the output variable, or there could be a complex non-linear relationship between the feature and output. Feature importance also does not imply a causal relationship between inputs and outputs.

Do not accept results at face value

We concluded from experiment 2 that a predictive signal for CAD case vs. control status prediction could be exploited by machine learning. The feature importance analysis of the gradient boosted tree suggested that `dob` is related strongly to case status. We can explore the relationship between dob and case status by computing the (future) case prevalence for each cohort of patients born in the same month (Figure 3). Surprisingly, this demonstrates a unique historical relationship between dob and eventual case prevalence, in which it appears that all patients born prior to 2001 presented as CAD cases, as did all patients born after 2013. Put differently, in this data set there were no controls born prior to 2001 or after 2013. This pattern seems likely to account for most of the importance of this feature.

Figure 3. Future case prevalence for patients separated into cohorts based on month of birth.



The information in Figure 3 should arouse suspicion that something has gone wrong, since this data is clinically unreasonable. Did database queries used to extract controls utilize a smaller date range than those that were used to extract cases? Is there a bug in the code used to generate the data in Figure 3? Is there some extraneous explanation for this pattern, such as changes in clinic population, marketing, or in the underlying database infrastructure? In fact, after some investigation we found that Figure 3 is an accurate representation of the case data used in this tutorial, but that code used to assemble the data set (which was mentioned but not part of this tutorial) contained a bug that inadvertently caused dates in the control group to be incorrectly converted, producing this artifact.

It is worth asking whether the machine learning models are extracting information from other features, or whether this artifact is essential to their apparent success. Therefore, it makes sense to remove `dob` from the feature set and see if there is predictive signal in the remaining features. Experiment 2 was then repeated, with the results shown in Table 6.

Table 6. Performance of machine learning models and a null baseline on the CAD case vs. control status prediction task, with `dob` removed from the feature set.

Model type	**Accuracy**	**F1-score**	**PR AUC**
Support vector machine	0.671 (0.651, 0.690)	0.444 (0.410, 0.477)	0.524 (0.495, 0.555)
Random forest	0.742 (0.723, 0.760)	0.534 (0.501, 0.566)	0.608 (0.576, 0.639)
Decision tree	0.705 (0.687, 0.723)	0.484 (0.449, 0.516)	0.561 (0.530, 0.592)
Bagging trees	0.733 (0.714, 0.751)	0.532 (0.498, 0.565)	0.603 (0.570, 0.632)
Boosting trees	0.735 (0.717, 0.752)	0.532 (0.497, 0.564)	0.604 (0.573, 0.635)
Logistic regression	0.708 (0.688, 0.727)	0.000 (0.000, 0.000)	0.646 (0.637, 0.655)
Null baseline	N/A	0.452 (0.431, 0.475)	0.646 (0.637, 0.655)

We observe a marked drop in the performance of all models, with most of the performance scores now dropping below those of the baseline classifier. For several of the tree-based ensemble approaches, the F1-score is significantly higher than baseline, but this is not true for the PR AUC. We conclude that a majority, if not all, of the predictive signal relevant to CAD case vs. control prediction is carried in the aberrant `dob` feature.

A lesson here is to not accept the initial results at face value. It is often advantageous to perform additional low-level analyses, such as to clarify the dynamics of potentially relevant signals, and to ensure that what we observe makes clinical and logical sense. In this case, the strength of the ML models' performance rested heavily on a single feature that was corrupted by data cleaning code upstream to our analysis.

Baselines are often critical for understanding system performance

Had we not included a null model in our experiments for baseline comparison, we might have been misled by relatively promising performance scores. It is important to remember that when there is class skew in the data set, not only is accuracy misleading, but other measures of performance are difficult to assess without a baseline model for comparison.

There are multiple choices for baseline classifiers. The indiscriminately-positive classifier is often considered the best for precision-recall analysis (F1-score and PR AUC), but is not an ideal baseline for accuracy.

Machine learning cannot solve every problem

One question that needs to be carefully addressed in any project is whether the training data actually carries signal relevant to the desired prediction task. Imagine trying to train a machine learning model to predict what you have for dinner based on an image of the sky above your house. You could supply all the training data you want – images of the sky, and what you had for dinner that day – but your ML classifier would do little good. (That is, unless what you eat for dinner actually does depend on what the sky looks like.) For machine learning to work, there must be a relationship between the inputs and the output. It is often not clear whether such a relationship exists, and one goal of ML might be to infer such relationships.

Success of data science initiatives depends on quality data management

Every veterinarian, practice owner, administrator, PIMS developer, and pet owner should advocate for the curation of high-quality animal health data. When data is well-structured, complete, and easily-accessible, it becomes a rich resource that in principle can improve quality of life by increasing our knowledge about a disease or improving the quality and efficiency of veterinary care.

This project demonstrates the importance of storing patient data in a direct, structured format. Although our CAD case definition considered a coded diagnosis, no patients were assigned a coded diagnosis of CAD. This is an example where structured data (using a particular diagnosis code rather than merely a free text description) could have improved our analysis if it were present in the medical record. Analysis of patient data is often significantly hindered by data being stored in the record as free text. If diagnoses were consistently and correctly coded, perhaps using a standard terminology like SNOMED-CT, it would open more possibilities for extracting value and insight.

Similarly, in our project treatment success was defined by the patient returning to the hospital at least one time for a repeat immunotherapy treatment. It would have been better to access a discrete problem list or other chart data that specifically encoded treatment success, since patient returns could be related to marketing or other demographic trends in the clinic population.

Machine learning models require data, but in veterinary clinical informatics applications the data sets we have access to are often relatively small. This dramatically increases the importance of having complete, direct, structured information like diagnoses and problem lists.

Be skeptical and communicate cautiously

The above subtitle is a warning to anyone reading or communicating about machine learning. Scientists usually recognize the importance of presenting their work without unnecessary fanfare, but it is easy for anyone to get carried away. When we learn about the newest, exciting developments in artificial intelligence, it may create a bias towards expecting machine learning to do unbelievable things. It is important to evaluate each new development with an open mind, and to recognize that a successful machine learning system depends as much on the specific task and the quality and quantity of available data as on the underlying methodology.

Conclusion

The central goal of machine learning is to design systems whose performance on some prediction task improves when given more data. To train a system to predict treatment success for canine atopic dermatitis (CAD), for example, we supplied it with instances of patients that did and did not experience treatment success. A learning algorithm will use this data to form a mathematical function that can predict treatment success using sex, breed, date of birth, and other patient features. We were unable to train a machine learning model to predict treatment success, and this could have been because the outcome is not strongly dependent on any of the available features.

We also trained machine learning models to predict CAD case vs. control status. They performed statistically better than baseline when the date of birth variable was included, but were indistinguishable from baseline when this feature was removed. The success of CAD case vs. control prediction was due to date of birth being corrupted by prior data processing, creating the appearance of an unusual historical relationship between date of birth and case status. Finally, we discussed common pitfalls and addressed data management issues relevant for people not directly involved in ML or data science work.

This toy example allowed us to delve into important concepts that are ubiquitous throughout the data science and machine learning literature. We hope that this tutorial helped to bring these ideas to life. There are many exciting, emerging technologies in veterinary medicine achieving positive results. Now, please consider how you might bring machine learning into your domain of expertise!

Acknowledgements

Many thanks to Rebecca White at the University of Wisconsin School of Veterinary Medicine (SVM) for her assistance extracting and assimilating medical record data. In addition, we are grateful to the 2018 Applied Epidemiology students and SVM staff and faculty who provided feedback on the initial work that led to this project. Thanks also to Mark Craven and Tony Goldberg for their critical input and mentorship.

Funding

NB was supported by the Computation and Informatics in Biology and Medicine Training Program (NLM 5T15LM007359) and the Comparative Biomedical Sciences Training Program (NIH T32OD010423).

References

1. Mitchell TM. Machine Learning. 1st edition. New York, NY: McGraw-Hill; 1997.

2. AI For Everyone. In: Coursera [Internet]. [cited 13 May 2020]. Available: https://www.coursera.org/learn/ai-for-everyone

3. Theobald O. Machine Learning For Absolute Beginners: A Plain English Introduction. 2 edition. Scatterplot Press; 2017.

4. James G, Witten D, Hastie T, Tibshirani R. An Introduction to Statistical Learning: with Applications in R. 1st ed. 2013, Corr. 7th printing 2017 edition. Springer; 2013.

5. Halliwell R. Revised nomenclature for veterinary allergy. Vet Immunol Immunopathol. 2006;114: 207–208. doi:10.1016/j.vetimm.2006.08.013

6. DeBoer DJ. The future of immunotherapy for canine atopic dermatitis: a review. Vet Dermatol. 2017;28: 25-e6. doi:10.1111/vde.12416

7. Jacobsen L, Valovirta E. How strong is the evidence that immunotherapy in children prevents the progression of allergy and asthma?: Curr Opin Allergy Clin Immunol. 2007;7: 556–560. doi:10.1097/ACI.0b013e3282f1d67e

8. Saridomichelakis MN, Olivry T. An update on the treatment of canine atopic dermatitis. Vet J. 2016;207: 29–37. doi:10.1016/j.tvjl.2015.09.016

9. Willemse A, Van den B, Rijnberk A. Effect of hyposensitization on atopic dermatitis in dogs. J Am Vet Med Assoc. 1984;184: 1277–1280.

10. Canonica GW, Cox L, Pawankar R, Baena-Cagnani CE, Blaiss M, Bonini S, et al. Sublingual immunotherapy: World Allergy Organization position paper 2013 update. World Allergy Organ J. 2014;7: 51. doi:10.1186/1939-4551-7-6

11. DeBoer DJ, Verbrugge M, Morris M. Clinical and immunological responses of dust mite sensitive, atopic dogs to treatment with sublingual immunotherapy (SLIT). Vet Dermatol. 2016;27: 82-e24. doi:10.1111/vde.12284

12. Mueller RS, Jensen-Jarolim E, Roth-Walter F, Marti E, Janda J, Seida AA, et al. Allergen immunotherapy in people, dogs, cats and horses – differences, similarities and research needs. Allergy. 0. doi:10.1111/all.13464

13. Bisgaard H, Halkjær LB, Hinge R, Giwercman C, Palmer C, Silveira L, et al. Risk analysis of early childhood eczema. J Allergy Clin Immunol. 2009;123: 1355-1360.e5. doi:10.1016/j.jaci.2009.03.046

14. Shams K, Grindlay DJC, Williams HC. What's new in atopic eczema? An analysis of systematic reviews published in 2009–2010. Clin Exp Dermatol. 2011;36: 573–578. doi:10.1111/j.1365-2230.2011.04078.x

15. Williams HC, Pembroke AC, Forsdyke H, Boodoo G, Hay RJ, Burney PGJ. London-born black caribbean children are at increased risk of atopic dermatitis. J Am Acad Dermatol. 1995;32: 212–217. doi:10.1016/0190-9622(95)90128-0

16. Torrelo A. Atopic dermatitis in different skin types. What is to know? J Eur Acad Dermatol Venereol. 28: 2–4. doi:10.1111/jdv.12480

17. Anturaniemi J, Uusitalo L, Hielm-Björkman A. Environmental and phenotype-related risk factors for owner-reported allergic/atopic skin symptoms and for canine atopic dermatitis verified by veterinarian in a Finnish dog population. PLoS One San Franc. 2017;12: e0178771. doi:http://dx.doi.org.ezproxy.library.wisc.edu/10.1371/journal.pone.0178771

18. Nødtvedt A, Bergvall K, Sallander M, Egenvall A, Emanuelson U, Hedhammar Å. A case–control study of risk factors for canine atopic dermatitis among boxer, bullterrier and West Highland white terrier dogs in Sweden. Vet Dermatol. 2007;18: 309–315. doi:10.1111/j.1365-3164.2007.00617.x

19. Sousa CA, Marsella R. The ACVD task force on canine atopic dermatitis (II): genetic factors. Vet Immunol Immunopathol. 2001;81: 153–157. doi:10.1016/S0165-2427(01)00297-5

20. Bizikova P, Santoro D, Marsella R, Nuttall T, Eisenschenk MNC, Pucheu-Haston CM. Review: Clinical and histological manifestations of canine atopic dermatitis. Vet Dermatol. 2015;26: 79-e24. doi:10.1111/vde.12196

21. Wilhem S, Kovalik M, Favrot C. Breed-associated phenotypes in canine atopic dermatitis. Vet Dermatol. 2011;22: 143–149. doi:10.1111/j.1365-3164.2010.00925.x

22. Nødtvedt A, Egenvall A, Bergval K, Hedhammar Å. Incidence of and risk factors for atopic dermatitis in a Swedish population of insured dogs. Vet Rec. 2006;159: 241–246. doi:10.1136/vr.159.8.241

23. Flohr C, Pascoe D, Williams HC. Atopic dermatitis and the ‘hygiene hypothesis’: too clean to be true? Br J Dermatol. 2005;152: 202–216. doi:10.1111/j.1365-2133.2004.06436.x

24. Johnson CC, Alford SH. Do animals on the farm and in the home reduce the risk of pediatric atopy?: Curr Opin Allergy Clin Immunol. 2002;2: 133–139. doi:10.1097/00130832-200204000-00009

25. Almqvist C, Garden F, Kemp AS, Li Q, Crisafulli D, Tovey ER, et al. Effects of early cat or dog ownership on sensitisation and asthma in a high-risk cohort without disease-related modification of exposure. Paediatr Perinat Epidemiol. 2010;24: 171–178. doi:10.1111/j.1365-3016.2010.01095.x

26. Meury S, Molitor V, Doherr MG, Roosje P, Leeb T, Hobi S, et al. Role of the environment in the development of canine atopic dermatitis in Labrador and golden retrievers. Vet Dermatol. 2011;22: 327–334. doi:10.1111/j.1365-3164.2010.00950.x

27. R Core Team. A language and environment for statistical computing. Vienna, Austria; 2018. Available: https://www.R-project.org/

28. Flach P, Kull M. Precision-Recall-Gain Curves: PR Analysis Done Right. In: Cortes C, Lawrence ND, Lee DD, Sugiyama M, Garnett R, editors. Advances in Neural Information Processing Systems 28. Curran Associates, Inc.; 2015. pp. 838–846. Available: http://papers.nips.cc/paper/5867-precision-recall-gain-curves-pr-analysis-done-right.pdf

29. Gao S, Young MT, Qiu JX, Yoon H-J, Christian JB, Fearn PA, et al. Hierarchical attention networks for information extraction from cancer pathology reports. J Am Med Inform Assoc. 2018;25: 321–330. doi:10.1093/jamia/ocx131

30. DiCiccio TJ, Efron B. Bootstrap Confidence Intervals. Stat Sci. 1996;11: 189–212.

31. Shalev-Shwartz S, Ben-David S. Understanding Machine Learning: From Theory to Algorithms. 1 edition. Cambridge University Press; 2014.

32. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine Learning in Python. J Mach Learn Res. 2011;12: 2825–2830.