I TRIED A BUNCH OF THINGS: THE DANGERS OF UNEXPECTED OVERFITTING IN CLASSIFICATION

MICHAEL SKOCIK[1], JOHN COLLINS[2], CHLOE CALLAHAN-FLINTOFT[3],HOWARD BOWMAN[4], AND BRAD WYBLE[3]

1. Manada Technology LLC

2. Physics Department, Penn State University

3. Psychology Department, Penn State University

4. Computing Department, University of Kent

See OSF page for code sample and data:
https://osf.io/qkvhd/?view_only=bb01fb14e61c405a936765f1524b36b9

ABSTRACT

Machine learning is a powerful set of techniques that has enhanced the abilities of neuroscientists to interpret information collected through EEG, fMRI, MEG, and PET data. With these new techniques come new dangers of overfitting that are not well understood by the neuroscience community. In this article, we use Support Vector Machine (SVM) classifiers, and genetic algorithms to demonstrate the ease by which overfitting can occur, despite the use of cross validation. We demonstrate that comparable and non-generalizable results can be obtained on informative and non-informative (i.e. random) data by iteratively modifying hyperparameters in seemingly innocuous ways. We recommend a number of techniques for limiting overfitting, such as lock boxes, blind analyses, and pre-registrations. These techniques, although uncommon in neuroscience applications, are common in many other fields that use machine learning, including computer science and physics. Adopting similar safeguards is critical for ensuring the robustness of machine-learning techniques.

INTRODUCTION

Computers have revolutionized approaches to data analysis in psychology and neuroscience, effectively allowing one to interpret not only the neural correlates of cognitive processes, but also the actual information that is represented in the brain through use of machine learning. However, with these new and powerful tools come new dangers. Machine learning algorithms allow a pattern classifier to weave many subtle threads of information together to detect hidden patterns in a massive data set, e.g. to determine from MEG data whether someone is currently

viewing a building or an animal (Cichy, Pantavis & Oliva 2014). However, these pattern classifiers are essentially black boxes to their human operators, as they create complex mappings between features and outputs that exceed one's ability to comprehend. Consequently, when using such algorithms, one can unintentionally overfit to the data, even when performing analyses that intuitively seem correct. This article seeks to provide a clear understanding of the dangers of overfitting by machine classifiers through example, and describes tools and methods to limit and detect it. We feel that a thorough understanding of the error introduced through overfitting is crucial, since this error is easy to commit yet difficult to detect. Furthermore, while there has been a lot of discussion of problems of circularity and inflated effects in neuroscience analyses (e.g. Kriegeskorte, Simmons, Bellgowan & Baker 2009; Vul, Harris, Winkielman & Pashler 2009; Eklund, Nichols, Anderson & Knutsson 2015; Brooks, Zoumpoulaki & Bowman, 2016), machine learning algorithms are so effective that they provide dangers above and beyond those that have been discussed. Importantly, as will be demonstrated below, *the technique of cross-validation, often employed as a safeguard against overfitting, is not always effective*. We suspect that the incidence of accidental overfitting errors in the literature could be substantial already, and may increase as machine learning methods increase in popularity.

Overfitting is the optimization of a model on a given data set such that performance improves on the data being evaluated but remains constant or degrades on other similar data. This 'other' data can be referred to as *out-of-sample*, meaning that it is outside of the data that was used to train and evaluate the classifier. In other words, if one were to develop a machine learning approach on one data set and then apply the same algorithm to a second set of data drawn from the same distribution, performance might be much worse than on the original set of data. This is a severe problem, because models that cannot generalize to out-of-sample data have little to say about brain function in general: Their results are valid only on the data set used to configure the classifier, may be significantly influenced by noise in the data, and are unlikely to be replicated on any other set.

Problems with overfitting are by no means new to science: High-energy physics has had a number of high-profile false discoveries, some of which were the result of overfitting an analysis to a particular data set. For accounts of some of these in the light of current experimental practice, see Harrison (2002) and Dorigo (2015). The similarities between data analysis in high-energy physics and modern neuroscience are striking: both fields have enormous quantities of data that need to be reduced to discover signals of interest. As such, it is useful and common to apply cuts to the data, i.e. to restrict analysis to certain regions of interest (ROI), as is common to the analysis of fMRI and EEG data. Because the purpose of the cuts is to enhance a signal of interest, there is a danger that the choice of a cut made on the basis of the data being analyzed (and on the basis of the desired result) may create a signal where none actually exists. Furthermore, when making measurements in high-energy physics and neuroscience, complicated apparatuses are often used, and analyses typically contain an extremely sophisticated set of

software algorithms. Optimization, and debugging of the analysis chain has the potential for creating an apparent measurement of a non-existent effect.

Because of several high-profile false discoveries, high-energy physics has already gone through a replicability crisis, and has had to rearrange its methods to deal with the consequences. A classic case, which was a big wake-up call, was the so-called split-A2 from Chikovani et al. (1967). Had this effect been genuine, it would have engendered a theoretical revolution, but when more data became available, the effect disappeared; see Harrison (2002) for a recent view. It appeared that inappropriate selection of data was the culprit, which is a form of overfitting.

To prevent such cases, the high-energy physics community has adopted several conventions and methods in the analysis and interpretation of data. Of these changes, the one that is most relevant to overfitting is called *blind analysis*, referring to a technique in which analysis optimization occurs without consulting the dependent variable of interest (e.g. Klein & Roodman 2005). Since the optimization algorithm is blind to the result of interest, overfitting is at least difficult, if not impossible. At the end of this paper we will discuss several preventative solutions, including blind analysis. Unlike physics, while related issues have been discussed in the literature (Kriegeskorte et al 2009), the neuroscience field has not yet fully responded to the dangers of overfitting when complex analyses are used, which increases the potential of false findings and presents a major barrier to the replicability of the literature.

CROSS-VALIDATION DOES NOT AUTOMATICALLY PREVENT OVERFITTING

In the neuroscience literature, a method that is typically employed to prevent overfitting is cross-validation, in which data are repeatedly partitioned into two non-overlapping subsets. In each iteration, classifiers are trained on one set and tested on the other and the results of multiple iterations are averaged together. Because the training and testing sets are disjoint in each iteration, the average performance on the test sets is believed to provide an unbiased estimate of classifier performance on out-of-sample data. However, such an estimate can only be believed as long as one important restriction is obeyed: After performing cross-validation, the analysis chain used to classify the data must not be modified to obtain higher performance *on that same data*. Reusing the same data to optimize analysis parameters can induce overfitting, *even if cross-validation is used at each iteration*. Overfitting can arise both from alteration of data preprocessing (e.g. artifact rejection thresholds, channel or voxel selection, feature weights, or other parameter modifications) or modifications to the classifier (e.g. kernel selection, statistical model used).

The reason that overfitting can occur despite cross validation is that all data sets are composed of an inseparable combination of signal and noise. The signal is the portion of the data containing the useful information that one would like the machine learning classifier to discover while the noise includes other sources of variability.. However, when an analysis is optimized over

multiple runs, the choice of hyperparameters can be influenced by how the noise affected the classification accuracy. Consequently, while the optimization improves classification accuracy on the data that are being tested, performance may remain constant or even worsen on out-of-sample data, because some portion of its noise is not shared with the in-sample data (Figure 1).

The inability of cross validation to prevent overfitting, is well known in the machine learning and machine vision communities (Domingos 2012), which are taking increasing care to avoid the problem. For example, machine-learning competitions on websites such as Kaggle.com provide contestants with sample data on which to optimize their models. The final evaluation of the contestants is performed on a different data set that is either held as confidential by the sponsoring organization, or is released only a few days before the end of the competition. Contestants who find ways to access the data more often than the rules permit are disqualified, their organizations can be barred from future competitions and in one recent high-profile case a lead scientist was fired (Markoff 2015).

In writing this paper, we share the experience of our colleagues in the physics and computer science disciplines so as to adopt more rigorous standards of machine learning before a new replicability crisis unfolds. It is not our intent to call out specific examples of bad practice in the literature, although in our survey of the neuroscience classification literature we found very few cases in which the appropriate precautions had clearly been used (i.e. some variety of blind or lock box analysis which will be described below). Without these precautions it is impossible to determine from a paper's methods whether overfitting occurred (unlike the identifiable double-dipping described by Kriegskorte, Simmons, Belgowan & Baker 2009). The difficulty in identifying cases of overfitting is that it would have occurred during the optimization of the analysis, and these steps are typically omitted from the methods. Another issue that we observe in the literature is that terminology is used inconsistently, which makes it hard to understand exactly what was done. To help clarify terminology, we offer a table describing common terms and descriptions of what they are typically taken to mean (Table 1). We suggest a new term, the Lock box, which refers to a set of data that is held-out from the optimization process for verification and should not be consulted until the methods have been completely determined. The term *held-out* data set is sometimes taken to mean this, but that term is used inconsistently and is easy to misinterpret. The term lock box more clearly indicates the importance of holding the data in reserve. More will be said about this below.

Next we provide clear examples of overfitting despite use of cross-validation using a sample of EEG data recorded from our own lab. We use real data here instead of simulated data, to ensure that the noise reflects the genuine variability typically found in similar datasets. In the described experiment, EEG data were collected while subjects tried to detect different kinds of targets. A support vector machine (SVM) classifier was used with various kernel function parameters to determine from EEG data alone what kind of target was present in a given trial. Hyperparameters were iteratively optimized through a genetic algorithm using a cross validation procedure.

Performance was compared to lock box data items that were set aside and not used in the genetic algorithm's fitness function. Performance was shown to improve on the data on which the classifiers were optimized, but not on the lock box data, which is a clear-cut demonstration of overfitting. Note that overfitting was relatively easy to evoke, despite the use of cross-validation. The obtained results, presented below, demonstrate that classifiers can easily be overfit to obtain performance that will not generalize to set-aside or out-of-sample data.

METHODS

EEG METHODS

Data from a previously collected, unpublished EEG data set were used for classification analysis. Since the aim of the paper is to demonstrate the misuse of analysis methods, only a brief description will be provided here, and readers can consult the supplementary methods for a more in-depth description. Data from 30 subjects were used, all of whom provided informed consent. Subjects were attempting to detect and identify targets in either of two streams of changing black letters, where targets could be either a colored letter (colored-singleton: CS) or black digit (Orthographic: Ortho). Data were collected from a 32-channel sintered Ag/AgC1 electrode array mounted in an elastic cap connected to a Neuroscan Synamps amplifier. Data were collected with a band pass filter of 0.05 to 100 Hz and were resampled to 250 Hz. Trials were rejected for excessive noise or eye movements. All ERPs were time locked to the first target onset. The baseline from a window -200 ms to 0 ms relative to T1 onset was subtracted from each trial.

STATISTICAL METHODS

USING NESTED CROSS-VALIDATION TO DEMONSTRATE OVERFITTING OF WEAKLY INFORMATIVE DATA

Data were selected to be potentially, but at most weakly informative of the target type. The 64 data points (256 ms) from each EEG channel following target onset were extracted from each trial and subjected to a Fast Fourier Transform (FFT). The log of the absolute value of the FFT was computed, and frequency bands across all leads were summed, resulting in 64 frequency values per trial. The goal of the classification would be to determine whether subjects had seen a CS or Ortho target within a given trial based just on these 64 frequency values that represented the scalp-wide power spectrum from the 256ms time period after target onset.

Nested cross-validation was used to evaluate overfitting. In this case, nested cross-validation is a procedure in which data is first divided into two sets: a parameter optimization set and a lock box. The parameter optimization set is used to optimize the model’s hyperparameters and thereby increase performance. The lock box is used to evaluate out-of-sample performance, and is not used for hyperparameter optimization. In the absence of overfitting, the model's

performance on the optimization set and lock box should be comparable at both the start and the end of the optimization process.

A support vector machine (SVM) classifier was used to classify the post-processed EEG data. The nested cross-validation procedure used a randomly-selected 85% of the trials for the parameter optimization set and the remaining 15% of the trials for the lock box. The entire parameter optimization set was then repeatedly partitioned into an inner training and inner testing set, and those inner sets were used in conjunction with a genetic algorithm in order to optimize feature weight vectors, which will be discussed below. All SVM classifiers used a radial basis function (RBF) kernel with $\gamma = 25$.

Optimization of the analysis was done by modifying a vector of 64 feature weights that were multiplied by the power values from the FFT analysis to allow the SVM classification to be focused on the most relevant frequency bands. At the start of the optimization procedure, 10 feature weight vectors were randomly constructed with 64 values ranging from 0.95 to 1.05. The parameter optimization set was then 10-fold cross-validated 10 times (i.e. the entire parameter optimization set was independently split ten times into inner training / inner testing sets consisting of 90% / 10% of data) for each feature weight vector, and the feature weight vector that produced the highest mean area under the curve (AUC) was declared the best vector. All other feature vectors were discarded, and nine new vectors were created by mutating the winner. This mutation process involved adding vectors of random numbers with amplitudes ranging from -0.05 to 0.05 to the best vector. This process was repeated 200 times to optimize the analysis.

To measure overfitting, after each cross-validation procedure on the parameter optimization set, the best feature weight vector was also used to classify the lock box. It should be noted that this was done merely to test for overfitting, and accessing a lock box multiple times can itself result in overfitting (if the results are used to influence analysis choices or stopping criteria). In this paradigm, only the results on the parameter optimization set were used to influence the optimization, as the genetic algorithm mutates the feature vector that performed best on the that set (irrespective of performance on the lock box).

To measure the statistical significance of the model's classification on the parameter optimization set, a permutation test was run after the final generation of the genetic algorithm. First, the analysis result was computed as the mean AUC across the ten inner training/testing partitions of the parameter optimization set (as above) using the final generation of feature weights. Then, all condition labels for the trials (i.e. the target-type) were randomly shuffled 100 times. After each such shuffling, for each of the ten partitions, the SVM classifier was retrained with the final feature vector and the AUC was measured. This process produced ten AUC values per shuffling, with one per partition. Then, for each shuffling, all ten AUC values were averaged together, producing one mean result per shuffling. This produced a null-hypothesis distribution of 100 mean AUC values. The p-value was then computed as the fraction of the null-hypothesis

distribution that was larger than the non-permutated classification result (i.e. the proportion of shufflings that produced a mean AUC greater than the mean AUC on the unshuffled data).

This entire nested cross-validation procedure was repeated independently 15 times to demonstrate the robustness of overfitting. In each case, the data were randomly repartitioned into a parameter optimization set and a lock box, and the genetic algorithm was used to optimize feature weights for the parameter optimization set over 200 generations. Results are presented separately for each of the 15 independent runs.

USING NESTED CROSS-VALIDATION TO DEMONSTRATE OVERFITTING OF NON-INFORMATIVE DATA

This analysis was repeated to demonstrate overfitting on entirely non-informative data. This was accomplished by using the same data and procedures as in the previous section, but randomly shuffling the target-type labels at the beginning of each analysis. This should result in classification failure (i.e. AUC = 0.50), and any result better than chance is indicative of overfitting.

OVERFITTING VIA KERNEL SELECTION

A final test was run to demonstrate that overfitting can occur by manipulating the hyperparameters of the SVM classifier, rather than feature weights. After partitioning data into a parameter optimization set and a lock box, different SVM kernel functions and parameters were used to classify the data. Mean performance through cross-validation of the parameter optimization set was compared with performance on the lock box. This was done for both the weakly-informative and non-informative data. For the weakly-informative data, a high correlation between results on the parameter optimization set and lock box would demonstrate genuine improvement in classifier performance while uncorrelated results would indicate overfitting. For the non-informative data, results better than chance (AUC = 0.50) would indicate overfitting.

Three different SVM kernels were used: radial basis function kernels k_r, polynomial kernels k_p, and sigmoidal kernels k_s. These kernel functions are given by

$$k_r = e^{-\frac{||x_1 - x_2||^2}{\gamma^2}}$$

$$k_p = (1 + c_2 x_1 \cdot x_2)^{\gamma}$$

$$k_s = \tanh(\gamma x_1 \cdot x_2 - C_0)$$

where x_1 and x_2 are the input feature vectors and γ and C_0 are free parameters. For k_r and k_p, γ was varied from 1 to 50 in increments of 1; for k_s, γ and C_0 were randomly selected from the range $[10^{-3}, 10^2]$, where values were evenly distributed across orders of magnitude. 50 different parameter sets were used for each kernel function. No genetic algorithm is used in this case.

RESULTS

The results on weakly-informative data are shown in Figure 2, which shows that performance improves on the parameter optimization set without similar changes on the lock box. This demonstrates that overfitting is occurring. The mean AUC at the start and end of the genetic algorithm process is listed in Table 2.

Feature weight optimization on non-informative data has a similar effect as shown in Figure 3. Without overfitting, performance should remain at levels of chance, i.e. AUC = 0.50. However, performance improved on all 15 runs, with a significant improvement relative to randomly shuffled data on all but one of the runs. The mean AUC at the start and end of the process is listed in Table 3.

The results of overfitting by kernel selection are shown in Figure 4. If overfitting is not occurring, and the kernel selection genuinely improves performance, then performance on the parameter optimization set and lock box should be highly correlated. The evident lack of correlation here shows that choosing the best-performing kernel on the parameter optimization set does not generalize to the out-of-sample data, which is a form of overfitting.

DISCUSSION

This paper demonstrates the ease with which overfitting can be induced when using machine learning algorithms despite the use of cross-validation. Any kind of iterative optimization of analysis hyperparameters is capable of causing overfitting. The approach used here is analogous to optimization procedures that are used in EEG classification such as discarding channels to improve classification performance. Our results show that if such selection is performed repeatedly on the same data then there is a substantial likelihood of overfitting, even with the use of cross-validation. Similar problems may exist with other hyperparameters, e.g. choosing time windows, frequency bands, or different ways of filtering out artifacts. This danger is present for both automated and manual parameter optimization.

Moreover, the same concerns apply to any kind of large neural data set. For example, in the case of using multi-voxel pattern-analysis (MVPA) on fMRI data, iterative optimization through selection of voxels or regions would lead to the same kinds of overfitting that we demonstrate here.

These results should not be taken to indict cross-validation as a poor methodological choice: It is considered to be state-of-the-art by many in the machine vision and machine learning communities for good theoretical and practical reasons (Arlot & Celisse 2010). However, our result does clearly indicate that cross-validation does not permit limitless analysis optimization.

There are several ways in which overfitting can be protected against, above and beyond cross-validation. We suggest that, in order to increase generalizability and replicability, journals publishing data from classification analyses encourage the use of one of the approaches listed below.

THE LOCK BOX APPROACH.

Setting aside a lock box of data makes it possible to determine whether overfitting has occurred. This entails setting aside an amount of data at the beginning of an analysis and not accessing that data until the analysis protocol is clearly defined, which includes all stages of pre-processing, artifact correction/rejection, channel or voxel selection, kernel parameter choice, and the selection of all other hyperparameters. This technique is already standard practice in machine vision competitions. When submitting a candidate for such a competition, the ultimate performance of the algorithm is evaluated on a separate set of data that is reserved until the final stage of the test. The workflow of using a lock box is shown in Figure 5.

We suggest that, moving forward, machine classification approaches to data analysis in neuroscience routinely incorporate a lock box approach, in which data are set aside at the beginning of the development of an analysis and not assessed until the paper is ready for submission. At this point, the data in the lock box should be accessed just one time (ideally) to generate an unbiased estimate of the algorithm's performance on the out-of-sample data. This result is likely to be less favorable than the data that were being used during optimization and should be published alongside the result from the cross-validation.

If it turns out that the results from the lock box test are unsatisfactory, a new analysis might be attempted, but if so, the lock box should be re-loaded with new data, either collected from a new sample or from additional data that were set aside at the beginning of the analysis (but not from a repartitioning of the same data that had originally been used). A possible alternative is to access the lock box multiple times during optimization, but to apply a correction to any resultant statistics as a function of the number of times the lock box data was evaluated.

Note that this lock box approach is evaluative. It does not prevent overfitting, but allows one to test whether it has occurred. However, the performance of the algorithm on the lock box is a non-overfit result.

THE BLIND ANALYSIS APPROACH

Blind analysis is an appropriate tool for preventing overfitting when testing a well-defined hypothesis. Under a blind analysis scheme, the input data is modified in a reversible way to prevent the optimization process from inadvertently becoming tuned to noise patterns in the data that are differential with regard to the contrast of interest. Some examples of using blind analysis include scrambling all condition labels and then adding some amount of noise to all signals (e.g. artificially increasing the amplitude of the P300 in one condition), or artificially adding 'target signals' to trials. The hyperparameters of the model can then be optimized to detect the features present in the modified data. Once the hyperparameters are locked in, the blind can be lifted (e.g. conditions unscrambled, noise removed), and the unbiased results can be calculated. The advantage of this approach is that all of the data can be used during the optimization phase, and the final evaluation of performance can be done across all of the data instead of just the outer-box set. Note that blind-analysis is a way to minmize overfitting. If used in conjunction with a lock box, one can both minimize and diagnose overfitting.

THE PRE-REGISTRATION APPROACH

Cross-validation does succeed in providing an unbiased estimate of out-of-sample performance when classification results are not used to iteratively optimize performance. Therefore, it is safe to pre-register a classification analysis before attempting it. Assuming that it is done in good faith, the pre-registration would provide evidence that the analysis hyperparameters were finalized prior to attempting the analysis using previously established methods. The advantage of this approach over the lock box is that all of the data can be used in the final estimate of performance. The disadvantage is that hyperparameter optimization is not permitted, which limits the success of the analysis.

CONCLUSION

The biggest danger of data science is that the methods are powerful enough to find apparent signal in noise, and thus the likelihood of data overfitting is substantial. Our results illustrate how easily overfitting can occur despite the use of cross validation. It can be difficult to detect overfitting without having an abundance of data, which can be costly to collect in the case of neuroscience. However, as reproducibility is a cornerstone of scientific research, it is vital that some method of assuring out-of-sample generalizability be used. By setting aside an amount of data that is not accessed until the absolute completion of model modification (i.e. a lock box), one can get an unbiased estimate of the generalizability of one's system and of how much overfitting has occurred. Alternatively, blind analysis methods and good faith pre-registrations of the analysis parameters minimize the possibility for overfitting to occur. Conversely, using any method that allows one to check performance on the same data repeatedly without independent data that has not been consulted can induce overfitting, inflating false positive results and damaging replicability. Some attention to these dangers at this point, when machine learning approaches in neuroscience are relatively nascent, will allow us to improve the state of science before inappropriate methods become standardized.

REFERENCES:

Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics surveys*, *4*, 40-79.

Brooks, J.L., Zoumpoulaki, A. & Bowman, H. (2016). Data-driven region-of-interest selection without inflating Type I error rate. Psychophysiology, in press.

Cawley, G. C., & Talbot, N. L. (2010). On over-fitting in model selection and subsequent selection bias in performance evaluation. *The Journal of Machine Learning Research*, *11*, 2079-2107.

Chikovani, G., Focacci, M. N., Kienzle, W., Lechanoine, C., Levrat, B., Maglić, B., ... & Grieder, P. (1967). Evidence for a two-peak structure in the A 2 meson. *Physics Letters B*, *25*(1), 44-47.

Cichy, R. M., Pantazis, D., & Oliva, A. (2014). Resolving human object recognition in space and time. *Nature neuroscience*, *17*(3), 455-462.

Domingos, P. (2012). A few useful things to know about machine learning.*Communications of the ACM*, *55*(10), 78-87.

Dorigo, T. (2015). "Extraordinary claims: the 0.000029% solution", EPJ Web of Conferences 95, 02003.

Eklund, A., Nichols, T., Andersson, M., & Knutsson, H. (2015, April). Empirically investigating the statistical validity of SPM, FSL and AFNI for single subject fMRI analysis. In *Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on* (pp. 1376-1380). IEEE.

Harrison, P.F. (2002). "Blind Analysis", J. Phys. G: Nucl. Part. Phys. 28, 2679-2691.

Klein, J. R., & Roodman, A. (2005). Blind analysis in nuclear and particle physics. *Annu. Rev. Nucl. Part. Sci.*, *55*, 141-163.

Kriegeskorte, N., Simmons, W. K., Bellgowan, P. S., & Baker, C. I. (2009). Circular analysis in systems neuroscience: the dangers of double dipping.*Nature neuroscience*, *12*(5), 535-540.

Kilner, J. M. (2013). Bias in a common EEG and MEG statistical analysis and how to avoid it. Clinical Neurophysiology, 124(10), 2062–3. http://doi.org/10.1016/j.clinph.2013.03.024.

Markoff (2015) Baidu Fires Researcher Tied to Contest Disqualification [Web log post], retrieved from http://bits.blogs.nytimes.com/2015/06/11/baidu-fires-researcher-tied-to-contest-disqualification/

Vul, E., Harris, C., Winkielman, P., & Pashler, H. (2009). Puzzlingly high correlations in fMRI studies of emotion, personality, and social cognition.*Perspectives on psychological science*, *4*(3), 274-290.

TABLES

Table 1. The terminology used in this and other papers are defined in this table.

Term	Definition
Machine Learning	Machine learning is the use of highly powerful mathematical tools to discern patterns in data. Typically, machine learning algorithms are trained with labeled data from two or more classes, and are then used to predict which class a new and unlabeled data item belongs to. Examples of machine learning algorithms are support vector machines (SVM) classifiers, random forest models, and naive Bayes classifiers.
Training and Testing sets	These terms generally refer to the two subsets of data used during cross-validation. However the term test-set is sometimes used to refer to data that has been set-aside for later evaluation. We advise against that usage for the sake of consistency.
Nested Cross-Validation	Nested cross-validation is the single partitioning of data into two sets: a parameter optimization set, and a lock box, for final evaluation. The parameter optimization set can then be repeatedly repartitioned into an inner training and inner testing set. Through this repartitioning, the model's parameters are optimized, and then an unbiased estimate of the performance of the model on out-of-sample data can be achieved by accessing the lock box. This method is still vulnerable to overfitting if the entire dataset is re-partitioned into new optimization and lock box sets.
Parameter Optimization Set	Used in a nested cross-validation scheme, a parameter optimization set is a partition of the data used to optimize model parameters. By optimizing only on the parameter optimization set, and by only checking performance once on the lock box set, one can obtain an unbiased estimate of model performance. Parameter optimization sets can be cross-validated (repeatedly and independently divided into inner training/testing sets) to train and test a model.
Lock Box	We provide the term lock box to mean a subset of data that are removed from the analysis pipeline at the very start of optimization and not accessed until all parameter adjustments and training have been completed. In this paper, the lock box is accessed several times: This is done merely to demonstrate overfitting,

	and doing so in practice could itself lead to overfitting to the lock box.
Hyperparameters	Often synonymous with parameters, hyperparameters refer to the constants that can be adjusted to modify model performance prior to the classification stage. Some common hyperparameters are feature weights, SVM kernel function parameters, and (especially in EEG) artifact rejection thresholds.

Table 2. The average AUC at the start and the finish of the parameter optimization process on weakly-informative data. If overfitting had not occurred, then performance would have been comparable at both the start and the end of the process for both the inner optimization and outer lock box sets.

	Starting AUC	Ending AUC
Inner	0.50 ± 0.02	0.59 ± 0.02
Outer	0.50 ± 0.05	0.48 ± 0.05

Table 3. The average AUC at the start and the finish of the parameter optimization process on non-informative data. Without overfitting, performance should remain around levels of chance, i.e. AUC = 0.50. However, performance improves on the inner optimization set, demonstrating that overfitting has occurred.

	Starting AUC	Ending AUC
Inner	0.50 ± 0.03	0.63 ± 0.03
Outer	0.52 ± 0.05	0.49 ± 0.06

FIGURES

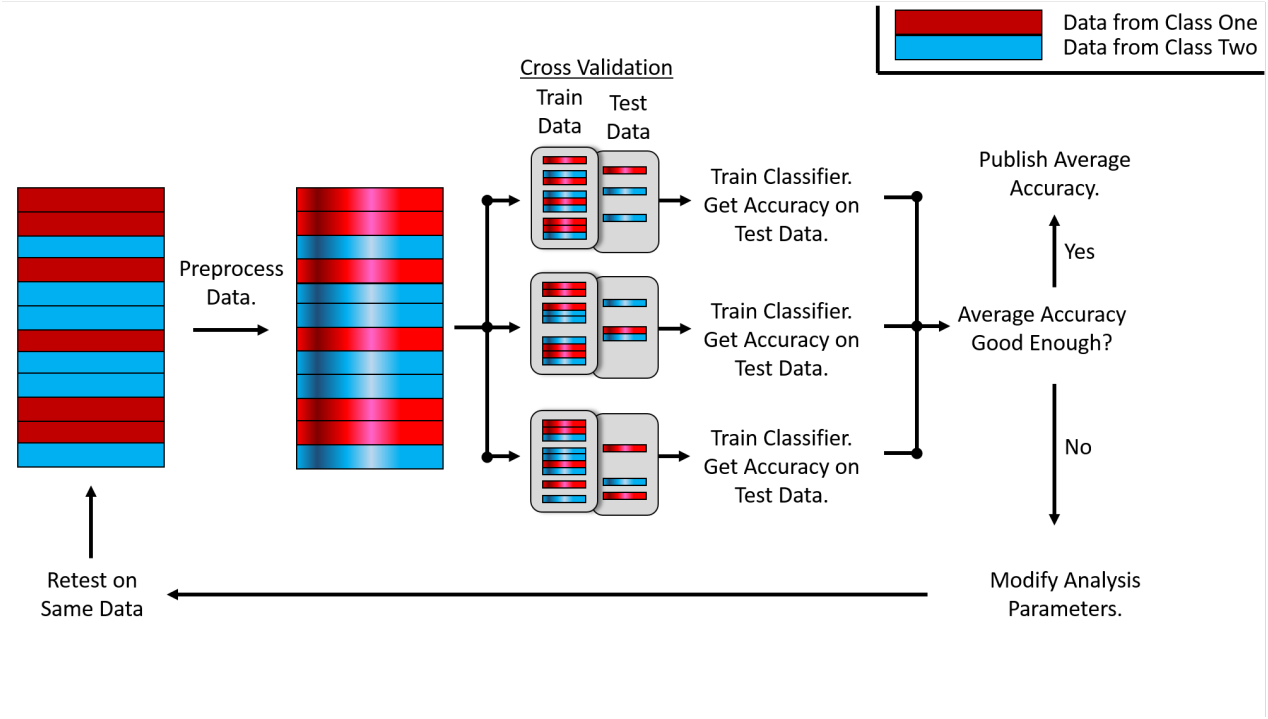


Figure 1. An example of how overfitting can be induced by modifying model hyperparameters after evaluating a system through cross-validation. It should be noted that overfitting can still occur even if, on each iteration, the same data is re-partitioned into unique training and testing sets, due to the feedback loop allowing hyperparameters to be adjusted.

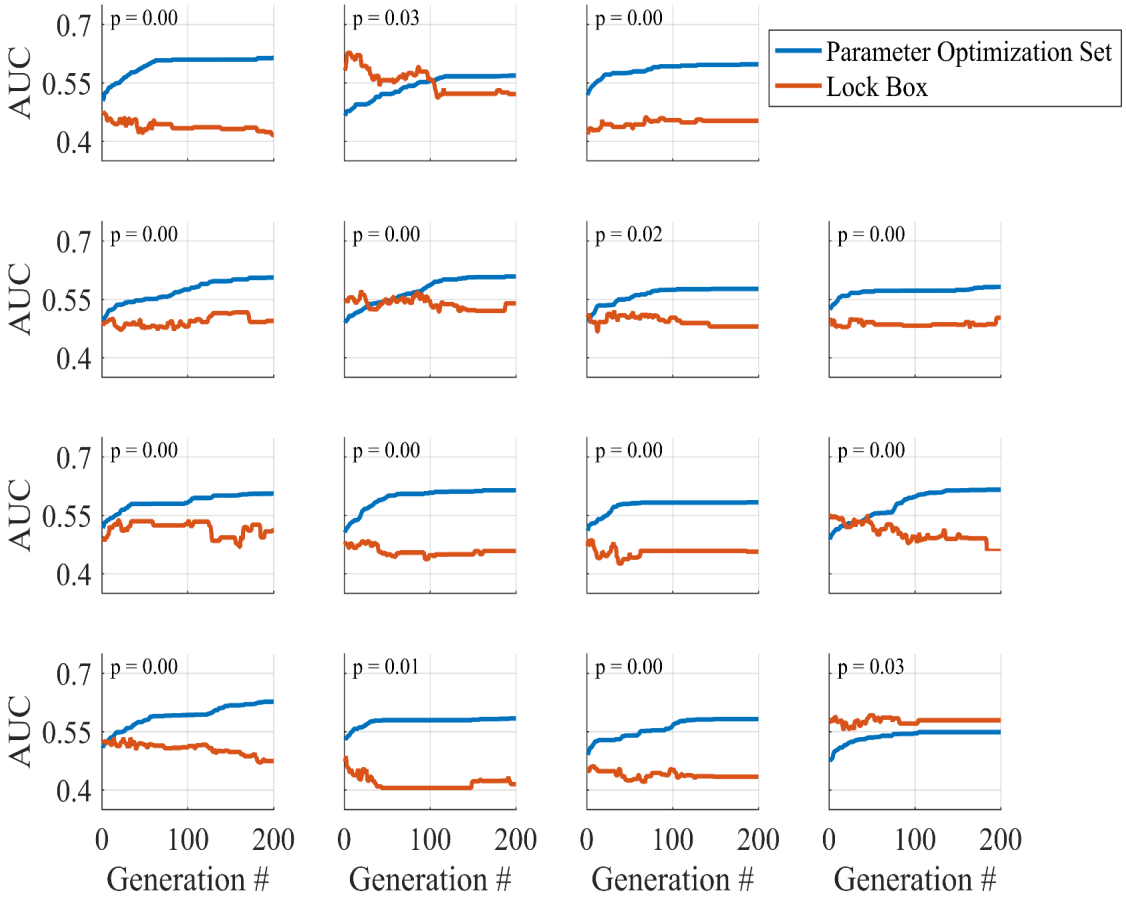


Figure 2. 15 tests of overfitting with cross-validation. EEG data is partitioned into a parameter optimization set and a lock box. A genetic algorithm is used to optimize feature weights over the parameter optimization set. Average area under the curve (AUC) on the parameter optimization set is reported each generation for the best feature weight vector (as computed through a ten-fold cross-validation procedure). To demonstrate overfitting, AUC is also computed each generation for the best feature weight vector on the lock box. If the model was optimizing to the signal, and not overfitting to the noise, performance would correspondingly rise for both the parameter optimization set and the lock box. P-values are computed for the parameter optimization set through a permutation test on the final generation of the genetic algorithm. (It should be noted that accessing a lock box multiple times can lead to overfitting. In practice, lock boxes should only be accessed at the end of the parameter optimization process.)

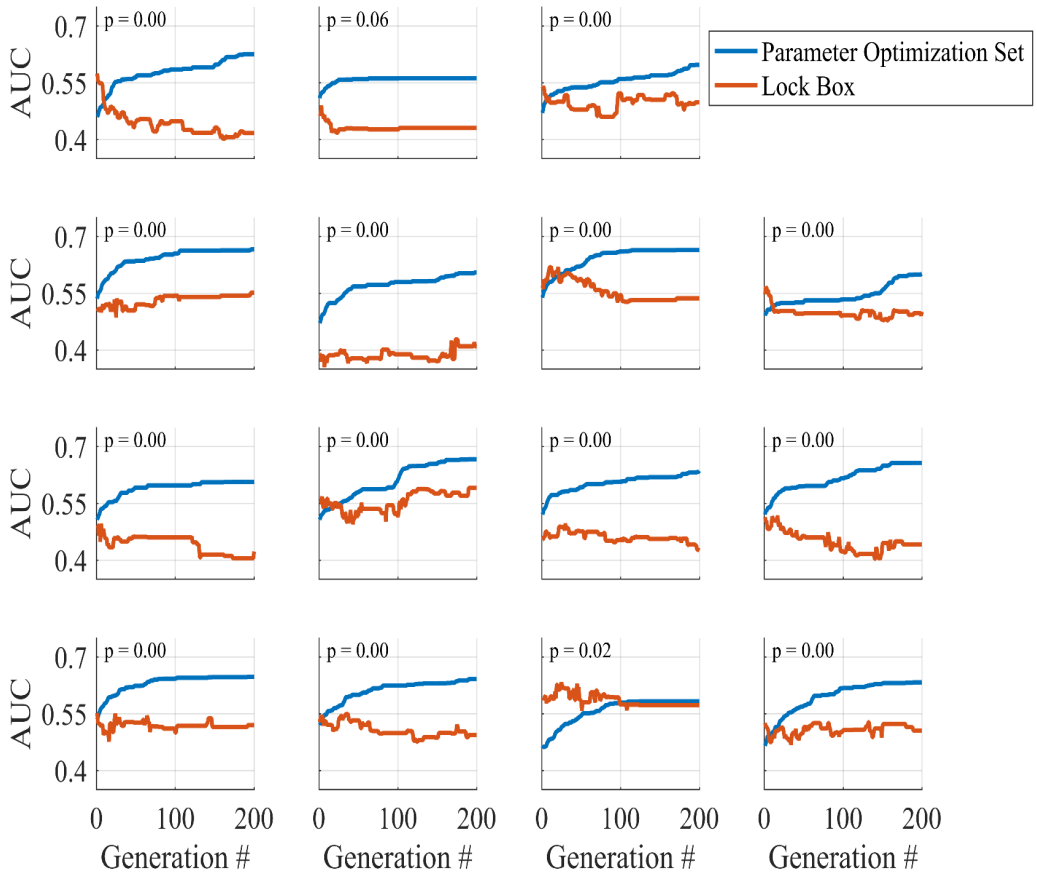
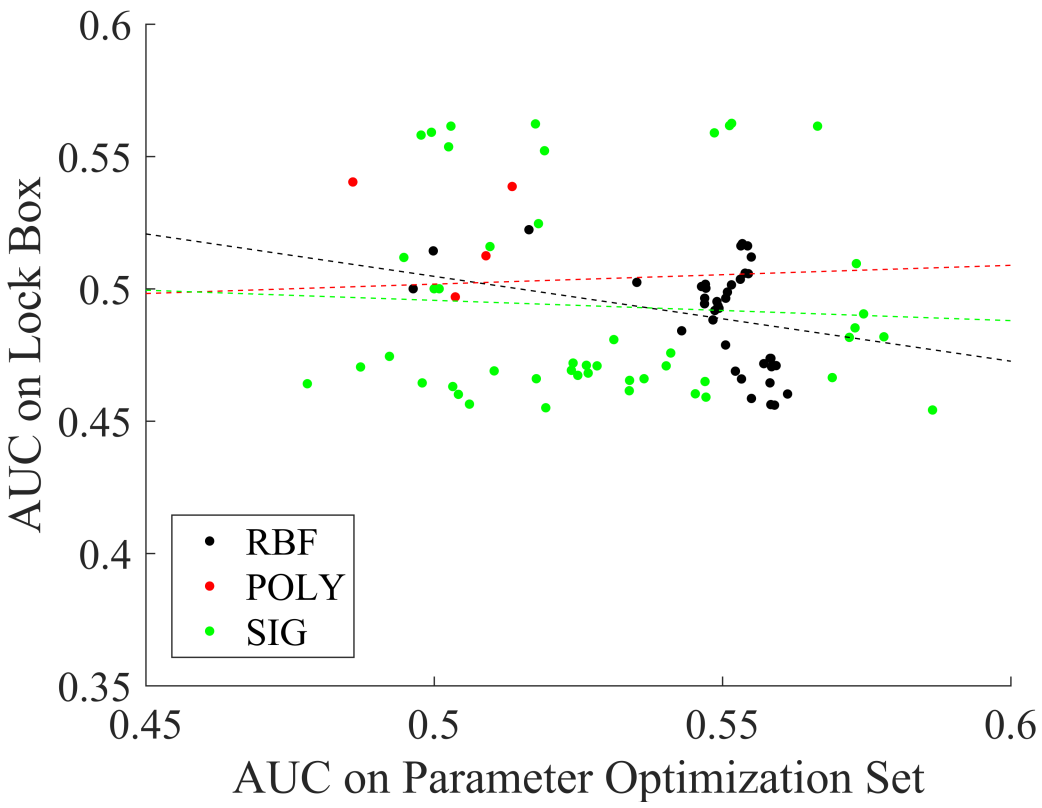
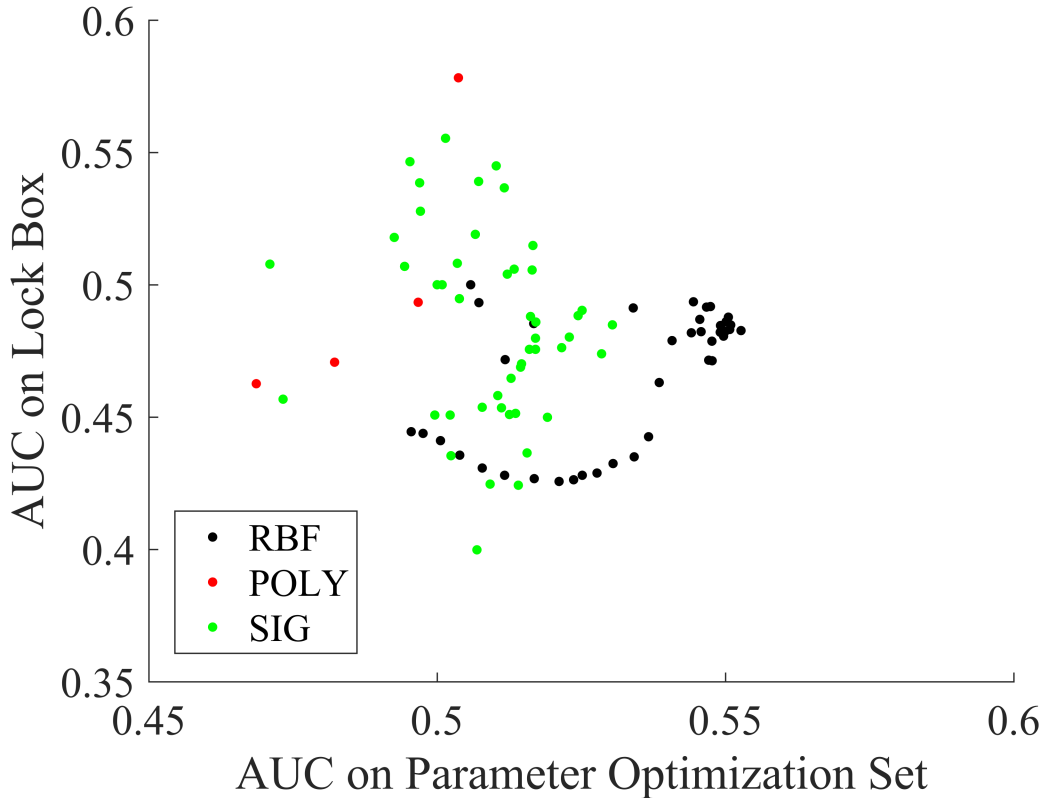


Figure 3. To demonstrate that models can be overfitted to non-informative data, the same test that was run in Figure 2 was run on EEG signals with randomly-assigned condition labels. Thus, the two classes the data was sorted into were (in a statistical sense) identical, and results should be at levels of chance (AUC = 0.50). However, through hyperparameter optimization, better-than-chance results can be achieved even on completely random data. P-values are computed for the parameter optimization set through a permutation test on the final generation of the genetic algorithm. (It should be noted that accessing a lock box multiple times can lead to overfitting. In practice, lock boxes should only be accessed at the end of the parameter optimization process.)



(a)



(b)

Figure 4. Choosing a classification model through trial-and-error can result in overfitting. Here, the data is partitioned into a parameter optimization set and a lock box, and three different kernel functions are used to classify the data: radial basis function (RBF), polynomial (POLY), and sigmoidal (SIG). The parameters for each of these kernel functions are modified, and average performance on the parameter optimization set is compared to performance on the lock box. (a) Performance on weakly-informative data. If overfitting had not happened, then results would be highly correlated. Lines of best fit are shown as dotted lines, indicating that there is little-to-no correlation between the results. (b) Performance on non-informative data. If overfitting had not happened, then all results would be both correlated and at the level of chance (AUC = 0.50). These tests demonstrate that model selection alone can induce overfitting.

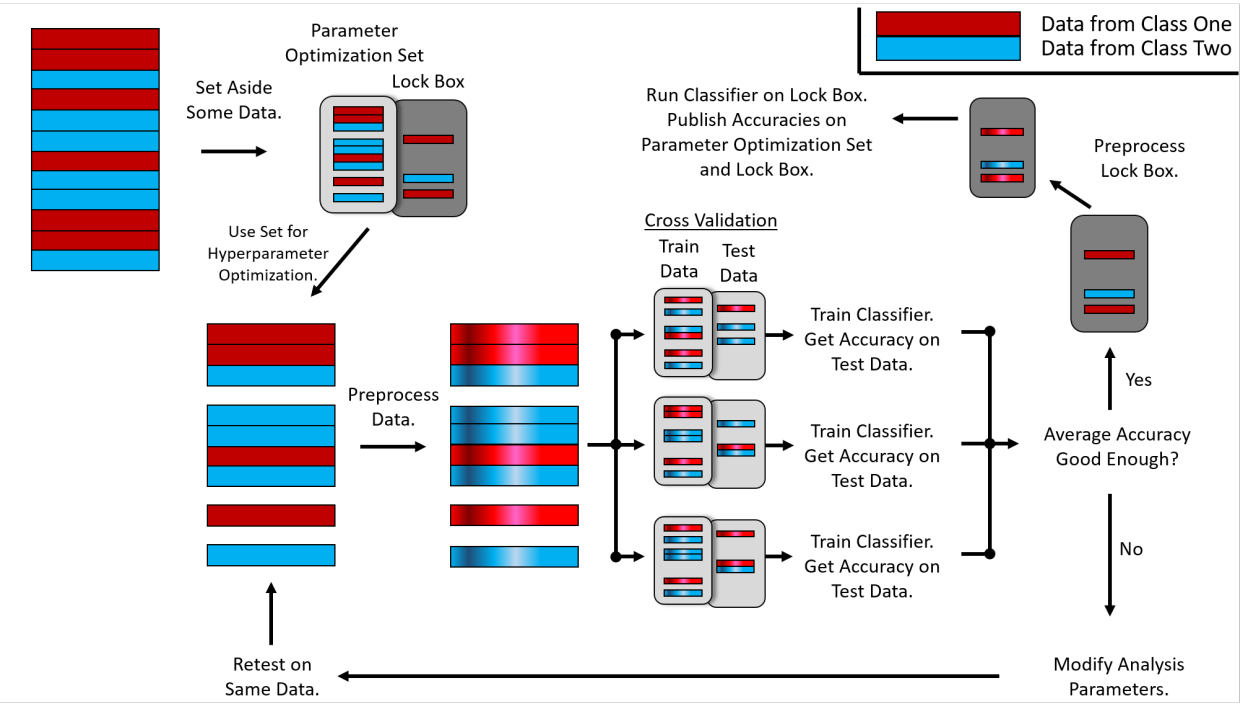


Figure 5. Here the workflow of using a lock box is demonstrated. Data is first divided into a parameter optimization set and a lock box. The model can be repeatedly tested and parameters can be iteratively modified on the parameter optimization set. After all hyperparameter optimization and model workflow is determined, the model can be tested against the lock box data. By doing this, an unbiased estimate of overfitting can be obtained, and an objective measure of how well this system will generalize to out-of-sample data is achieved.