Classification using Predictive State Smoothing (PRESS): A scalable kernel classifier for high-dimensional features with variable selection

Georg M. Goerg (gmg@google.com)

Google

March 15, 2018

Abstract

In this work we adapt the predictive state smoothing (PRESS) framework to classification, which leads to a fully probabilistic, non-linear classifier that estimates the minimal sufficient statistic for predicting class membership probabilities. It can be used for high-dimensional problems, both in number of observations and covariates, and allows for variable selection using LASSO and Ridge penalties. Out-of-sample prediction performance is comparable to existing state-of-the-art classifiers on several benchmark datasets; yet a trained PRESS classifier provides meaningful domain-specific insights based on regression coefficients using standard frequentist as well Bayesian inference. We also formulate PRESS as an equivalent state-dependent neural network representation. Algorithms scale linearly in the number of observations, can be optimized in batch mode, and can be easily implemented in R, STAN, or TensorFlow.

Keywords: kernel classification, predictive states, non-parametric smoothing, minimal sufficient statistic, distribution clustering, variable selection, high-dimensional data, dimension reduction, probabilistic neural nets.

1 Introduction

Recently Goerg (2017) introduced *predictive state smoothing (PRESS)* – a fully probabilistic framework for estimating minimal sufficient predictors. For the regression problem of predicting response $\mathbf{y}$ from features, PRESS can be viewed as a metric learner that estimates the smoothing matrix $\mathbf{S}$ from p-dimensional features $\mathbf{x} \in \mathcal{X}$ to get smoothed response $\hat{\mathbf{y}} = \mathbf{S} \times \mathbf{y}$. It achieves this by mapping features to a latent predictive state space $\mathcal{S}$, $\epsilon : \mathbf{x} \mapsto S$, which then constructs the kernel matrix via an inner product. This state space is constructed in such a way that it is *minimal sufficient* to predict y (Lauritzen, 1974; Dawid, 1979; Shalizi, 2003). It makes y independent of $\mathbf{x}$ given its state, $P(y \mid \mathbf{x}, \epsilon(\mathbf{x})) = P(y \mid \epsilon(\mathbf{x}))$, and it does so in a way that achieves maximal compression of $\mathbf{x}$ while not losing any information to predict y. This leads to a smoothing method that learns a generative model for $\mathbf{S}$ from $\mathbf{x}$.

It is important to note that the predictive state framework is fully probabilistic and does not require continuous response y. Hence in this work we adapt PRESS to the multi-label classification setting to estimate class membership probabilities

for $y \in \{1, \ldots, K\}$ conditioned on features $\mathbf{x}$.[1] Just as for regression, we show that a PRESS classifier achieves predictive performance on a par with state-of-the-art machine learning approaches such as Random Forests or support vector machines (SVMs) (Breiman, 2001; Cortes and Vapnik, 1995; Hearst, 1998); yet it is a generative model that conforms to laws of probability and hence can be used as building blocks in other probabilistic models. PRESS models usually have a lower dimensional parameter space compared to deep neural nets (Schmidhuber, 2015) or Random Forests – at comparable predictive performance –, which leads to smaller computing and memory requirements for estimation and storage of trained models. Moreover, it is also easy to interpret using parametric inference familiar from generalized linear models; it even lends itself naturally to a Bayesian inference approach.

The main contributions of this work are threefold: a) extend PRESS to the case of Bernoulli and Multinomial response; b) embed PRESS into a standard neural network structure with specific requirements for layers and activation functions; c) a fully Bayesian inference approach.

Section 2 adapts the predictive state framework from PRESS regression to the (multi-label) classification setting and gives a review of related work. In Section 3 we present algorithms for maximum likelihood estimation (MLE) as well as Bayesian inference of the state space ($\mathcal{S}$) and the mapping (ϵ) from features to states. In Section 4 we apply PRESS to several benchmark datasets and show that it does not only match or even outperform state-of-the-art classifiers, but also gives meaningful domain-specific insights from the estimated state space and ϵ mapping. Section 5 summarizes the methodology.

[1] All of the terminology, methodology, interpretation for predictive states for classification carries over exactly from the regression setting. Hence unless any classification specific insights are noteworthy we refer the reader to Goerg (2017) for details.

2 Predictive States for Classification

For sake of simplicity, consider a binary class label $y \in \{0, 1\}$,[2] which we model as a Bernoulli random variable with success probability $p \in [0, 1]$ and pdf

$$P(y) = p^y \cdot (1 - p)^{(1-y)} . \tag{1}$$

For the purpose of this work we consider classification problems where estimating well-calibrated class membership probabilities is important – rather than only a good label prediction. In that case one is interested to learn how the probability of success changes with additional information from features $\mathbf{x} \in \mathcal{X}$

$$P(y \mid \mathbf{x}) = p(\mathbf{x})^y \cdot (1 - p(\mathbf{x}))^{(1-y)} . \tag{2}$$

A common approach is then to estimate $p(\mathbf{x})$ using, e.g., a linear model on logit scale for logistic regression or a deep neural network that maps inputs $\mathbf{x}$ via a convolution of several linear combination layers to an output $\hat{p} = \text{sigmoid}(\hat{f}_{weights}(\mathbf{x}))$, where the sigmoid activation function guarantees that $\hat{p} \in (0, 1)$.

PRESS, on the other hand, does not assume a deterministic functional relationship $p(\mathbf{x}) =$

[2] All of the presented methodology, derivations, and algorithms apply without modification to multi-label classification (Multinomial distribution) by viewing it as a multivariate boolean response using a one-hot encoding of the class label. For example, for $K = 3$ classes represent label $i = 1, 2, 3$ as the 3-dimensional $y = e_i$, where e_i is a row-vector with a 1 in the i-th position, and 0 otherwise, e.g., $e_1 = (1, 0, 0)$.

sigmoid($f(\mathbf{x})$), but obtains the optimal predictive representation to predict y from $\mathbf{x}$ using latent states $S \in \mathcal{S}$,

$$\epsilon : \mathcal{X} \mapsto \mathcal{S}, \quad S = \epsilon(\mathbf{x}). \tag{3}$$

The state space is constructed in such a way that ϵ maps $\mathbf{x}$ to an equivalence class of all $\tilde{\mathbf{x}} \in \mathcal{X}$ that have the same predictive distribution as $\mathbf{x}$. Formally,

$$\epsilon : \mathbf{x} \mapsto [\mathbf{x}] = \{\tilde{\mathbf{x}} \mid P(y \mid \mathbf{x}) \equiv P(y \mid \tilde{\mathbf{x}})\}. \tag{4}$$

The set $[\mathbf{x}]$ is non-empty as it contains at least $\mathbf{x}$ itself. As shown in Shalizi (2003); Goerg (2017) $\epsilon(\mathbf{x})$ is the minimal sufficient statistic for predicting y from $\mathbf{x}$, $P(y \mid \epsilon(\mathbf{x}), \mathbf{x}) = P(y \mid \epsilon(\mathbf{x}))$. This is key for prediction and estimation as

$$\begin{aligned} P(y \mid \mathbf{x}) &= \int_{s \in \mathcal{S}} P(y \mid s, \mathbf{x}) P(s \mid \mathbf{x}) \mathrm{d}s & (5) \\ &= \int_{s \in \mathcal{S}} P(y \mid s) P(s \mid \mathbf{x}) \mathrm{d}s, & (6) \end{aligned}$$

where (6) follows from conditional independence of y and $\mathbf{x}$ given the (minimal) sufficient $\epsilon(\mathbf{x})$.

As for PRESS regression we use a finite state space $\mathcal{S} = \{s_1, \ldots, s_J\}$ that forms the basis of a continuous, probabilistic state space, where each s_j is a a basis or *extremal* state. In this case, (6) reduces to a mixture distribution

$$\begin{aligned} P(y \mid \mathbf{x}) &= \sum_{j=1}^{J} P(y \mid s_j) \cdot P(s_j \mid \mathbf{x}) \\ &= \sum_{j=1}^{J} w_j(\mathbf{x}) \cdot p_{s_j}, & (7) \end{aligned}$$

where $p_{s_j} := P(y \mid s_j)$ are state-conditional distributions and $w_j(\mathbf{x}) := P(s_j \mid \mathbf{x})$. Each weight vector $\mathbf{w}(\mathbf{x}) = (w_1(\mathbf{x}), \ldots, w_J(\mathbf{x}))$ lies in the J-dimensional probability simplex

$$\Delta^{(J)} := \{\mathbf{p} \in \mathbb{R}^J \mid p_j \geq 0 \text{ and } \sum_{j=1}^{J} p_j = 1\}, \tag{8}$$

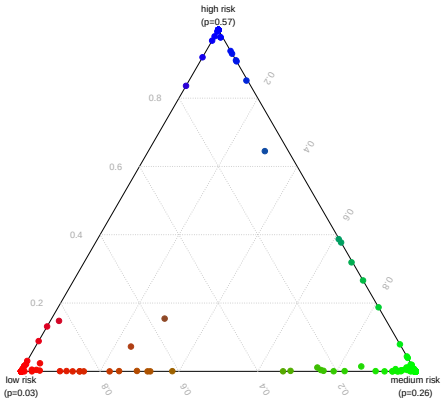
and represents the *probabilistic predictive state* of $\mathbf{x}$. A sample of N observations, $\mathbf{X} \in \mathbb{R}^{N \times p}$, can be represented as an $N \times J$ matrix $\mathbf{W}$ with

$$[0, 1] \ni \mathbf{W}_{i,j} = P(s_j \mid \mathbf{x}_i), \tag{9}$$

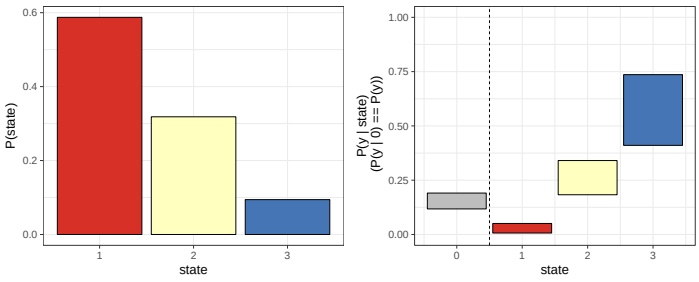
with row sums adding up to 1 as they represent a probability distribution over J states. The column sums, $\sigma_j = \sum_{i=1}^{N} \mathbf{W}_{i,j}$, represent number of samples in state s_j and satisfy $\sum_{j=1}^{J} \sigma_j = N$.

Predictive states in practice: As a motivating example of the applicability and interpretation of (probabilistic) predictive states consider the problem of assessing the risk of a life-threatening surgery. Doctors might characterize a patient as a "low", "medium", or "high" risk case with an associated chance of survival $\{p_{low}, p_{medium}, p_{high}\}$. For an individual patient's survival chance, all one needs to know is where their health history lies in the 3-dimensional state space $\mathcal{S} = \{$"low", "medium", "high"$\}$.

From a standard statistical machine learning point of view, this categorization usually occurs given model predictions, i.e., patients are labeled as "low" / "medium" / "high" risk *after* estimating $p(y \mid \mathbf{x})$ using a classifier and bucketing fitted probabilities in low / medium / high intervals, e.g., terciles. PRESS, on the other hand, uses the domain-specific inference as a natural – *latent* – component of the model, i.e., "low", "medium", "high" risk patients are not an *outcome* of a completely unrelated classifier (logistic regression, SVMs, random forest, neural nets, etc.), but are an inherent *building block* of the model in that they exactly represent the predic-



(a) Predictive state space with $J = 3$ for the surgery dataset (Section 4.1). Patient i is represented by $\mathbf{w}_i \in \Delta^{(3)}$ (the color of each dot is the RGB value of $\mathbf{w}_i$).



(b) Estimates of the marginal probability of each state $p(s_j)$ as well as marginal and state-conditional label distributions, $p(y)$ and $p(y \mid s_j)$.

Figure 1: Surgery dataset: PRESS estimates for predictive state space ($J = 3$) and associated optimal predictive distributions.

tive states that are sufficient to inform the probability of survival.

After learning how patients health history – features like tumor size or whether they smoke – map to the {"low", "medium", "high"} state space, the probability of survival for an individual patient, $p(y_i \mid \mathbf{x}_i)$, can be computed using a weighted average across the survival probabilities for each state according to (7). This does not only yield an efficient data reduction from p covariates $\mathbf{x}_i$ to a 3-dimensional state space $\mathbf{w}_i$, but also yields domain-specific interpretable insights on what contributes to being a "low" / "medium" / "high" risk patient, which is often difficult or impossible to obtain with alternative non-linear, non-parametric methods.

Figure 1a shows an estimated $J = 3$ dimensional state space, with corners of the triangle representing the *extremal* states s_{low}, s_{medium}, and s_{high}, respectively. Each patients health history $\mathbf{x}_i$ has its probabilistic state representation $\mathbf{w}_i$ represented as points on the simplex.

Figure 1b shows the (relative) size of each state ($\hat{p}(S = s_j) = \sigma_j/N$) and the state-conditional distributions $p(y \mid s_j)$. State 3 is the smallest state, yet if a patient is close to state 3 they have a high risk of complications after the surgery ($p(y \mid s_{high}) = 0.57$).

2.1 Bernoulli random variables as latent state level

Due to sufficiency of s_j

$$p(y \mid \tilde{\mathbf{x}}) = \sum_{j=1}^{J} P(S = s_j \mid \tilde{\mathbf{x}}) p(y \mid s_j), \qquad (10)$$

is a weighted average of state-conditional predictive distributions $p(y \mid s_j)$, which – by construction – are again Bernoulli with success probability p_{s_j}. Hence one can view a PRESS classifier as a multilevel hierarchical model with a latent state-

conditional level,

$$y_i \mid \mathbf{x}_i \sim \text{Bernoulli}(p(\mathbf{x}_i)) \tag{11}$$

$$y_i \mid s_j \sim \text{Bernoulli}(p_{s_j}), \quad j = 1, \ldots, J, \tag{12}$$

$$p(\mathbf{x}_i) = \mathbf{w}(\mathbf{x}_i) \times \mathbf{p}_S, \tag{13}$$

$$\mathbf{w}(\mathbf{x}_i) \sim \text{Dirichlet}(\mathbf{x}_i^\intercal \boldsymbol{\beta}), \tag{14}$$

where $\mathbf{p}_S = (p_{s_1}, \ldots, p_{s_J}) \in [0,1]^J$, p_{s_j} is the probability of success when in state s_j, and $\boldsymbol{\beta}$ are regression coefficients that parametrize the mapping from p features to J states (assuming a linear ϵ mapping).

Predictive states in practice (cont'd): To understand what contributes to a higher risk, it is useful to examine the coefficients that parametrize the (linear) mapping from features to states $\boldsymbol{\beta}$ in (14). The estimated $\widehat{\boldsymbol{\beta}}$ in Figure 2 shows that it's more likely for a patient to be at "high" risk if – as expected – they have a large tumor size or if they smoke. See Section 4.1 for more details.

2.2 Neural network interpretation

Originally derived from first principles of statistical inference for optimal probabilistic prediction,

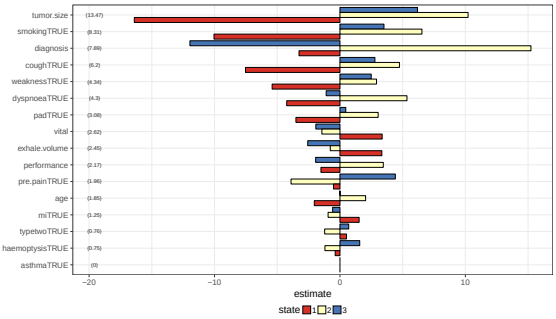


Figure 2: Coefficient estimates for linear mapping of p features to J states. Features are ordered from top to bottom, sorted by decrease in importance (value on left in parenthesis). See (24) for details.

PRESS can also be thought of as a metric learner for a smoothing matrix (Goerg, 2017). In its simplest form, the metric learner is parametrized using a linear relationship between features $\mathbf{x}$ and $\Delta^{(J)}$, and the smoothing matrix is constructed using an inner product on the predictive state space.

The neural network point of view is another way to understand and interpret PRESS, as (7) can be written as

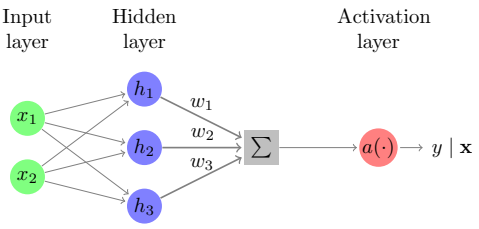
$$\text{output}_i = \langle \epsilon\left(\text{input}_i\right), \text{a}(\nu) \rangle, \tag{15}$$

where $\text{a}(\cdot)$ is an activation function, ν is a free parameter satisfying $\nu := a^{-1}(\mu)$ by construction, and $\langle \cdot, \cdot \rangle$ is the dot product. The two components are:

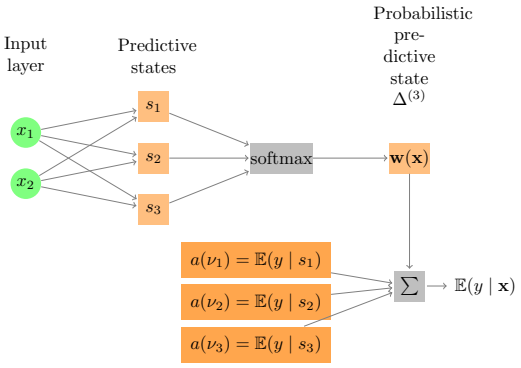
ϵ **mapping:** This can be easily generalized to any neural net classifier with J nodes and a "softmax" activation for the last layer ($\text{DNN}_{\text{softmax}}$ with all node weights represented as a high-dimensional parameter θ). In this case one can think of ϵ as simply a feature transformation step that transforms input features through several layers to the predictive states.

state-conditional mean $\mathbb{E}(y \mid s_j)$: If state-conditional means $\mu = a(\nu) \in \mathbb{R}^{J \times k}$ are considered free parameters, this is a $J \times k$ dimensional weight matrix. The type of activation function depends on the problem (e.g., sigmoid for classification), with the restriction that it must be closed under convex combination as activations for each state are averaged over $\mathbf{w}_i = \epsilon(\mathbf{x}_i)$. For example, identity, sigmoid, and softmax are all closed under convex combinations.

It is important to point out that the activation occurs *inside* the dot product, not outside, i.e., a weight constraint, not a kernel activation. This



(a) A standard dense neural network architecture with 1 hidden layer (3 nodes) and a univariate output $y = a\left(b + \sum_i w_i \cdot h_i(\mathbf{x})\right)$.



(b) Architecture of a PRESS smoother for $y = \sum_{j=1}^{J} w_j(\mathbf{x}) \cdot a(\nu_j)$ with a $J = 3$-dimensional predictive state space. An equivalent way to represent this as a standard DNN (left) is to impose activation function constraints on the edge weights, i.e., $w_i \rightsquigarrow a(w_i)$.

Figure 3: Comparison of a standard dense neural network with the analogous PRESS version. Instead of mapping from input to outputs directly (left), PRESS maps to the probabilistic predictive state space $\Delta^{(J)}$ and then estimates $\mathbb{E}(y \mid \mathbf{x})$ conditioned on the – minimal sufficient – state space. The conditional distributions (expectations) can be estimated separately as free parameters $a(\nu_j)$ or are inferred as weighted MLE from training data y and weights $\mathbf{W}$.

also hints at an interesting "role switch" for edges and nodes in a standard NN versus PRESS. Figure 3 illustrates these differences as both architectures model the output y as a linear combination of values: in a standard dense DNN, nodes represent predictions that are combined using edge weights; in PRESS, roles are reversed, as nodes *are* the weights (obtained via the "softmax" activation), which are used to combine prediction estimates. Figure 3 shows the state-conditional predictions as separate nodes, $a(\nu_j)$, for a more intuitive illustrations. However, one can equivalently represent PRESS with the same architecture in Figure 3a, but where edge weights from the last state are constrained by the activation function, i.e., weights equal $a(v_i)$.

The rationale is that contrary to standard DNNs, in PRESS inputs flow to the state space and the output prediction is a weighted average over the state space. Conceptually this aligns better with a probabilistic view of the world, rather than the deterministic view in standard DNNs that maps inputs to outputs directly.

In practice, we observe that this leads to faster convergence as the optimization can separate optimizing the low-dimensional μ to give good state-conditional estimates from the harder problem of estimating the much higher-dimensional θ that maps to the bounded predictive state simplex. Moreover, if ν is initialized to $\mathrm{a}^{-1}(\bar{\mathbf{y}})$ ($\bar{\mathbf{y}}$ being the sample mean) for each state, then $\hat{y}_i \equiv \bar{\mathbf{y}}$ for each i – for any initial θ_0 in $\mathrm{DNN}_{\mathrm{softmax}}$. In practice, this results in substantial convergence speedups compared to standard multi-level DNNs that have to map inputs to outputs directly and thus are not guaranteed to be at least as good as the sample mean at initialization.

2.3 Kernel smoothing

The state-conditional p_{s_j} can be estimated using the maximum likelihood estimator (MLE), which

is a weighted average of labels in each state,

$$\hat{p}_{s_j} = \frac{1}{\sigma_j} \sum_{i=1}^{N} \mathbf{W}_{i,j} \cdot y_i, \tag{16}$$

and the point prediction for observation i, $\hat{p_i}$, is a weighted average of state-conditional estimates. As for regression, this leads to a linear smoother with a closed form expression for the in-sample probability fit

$$[0,1]^N \ni \hat{\mathbf{p}} = \mathbf{S}(\mathbf{W}) \times \mathbf{y}, \tag{17}$$

$$[0,1]^{N \times N} \ni \mathbf{S}(\mathbf{W}) = \mathbf{W} \times \mathbf{D}(\mathbf{W}) \times \mathbf{W}^\intercal, \tag{18}$$

where $\mathbf{D}(\mathbf{W}) \in \mathbb{R}^{J \times J}$ is a diagonal matrix with $\mathbf{D}_{j,j} = \sigma_j^{-1}$.

While PRESS methodology builds on the $N \times N$ kernel matrix $\mathbf{S}$, it can rely on the kernel trick (Hofmann et al., 2008) and does not ever need to compute it explicitly, but predictions can be obtained in two steps with each scaling linearly in N: i) estimate state-conditional predictions, $\bar{\mathbf{p}}_S = \mathbf{D}(\mathbf{W}) \times \mathbf{W}^\intercal \times \mathbf{y} \in [0,1]^J$; ii) average over state predictions, $\hat{\mathbf{p}} = \mathbf{W} \times \bar{\mathbf{p}}_S \in [0,1]^N$.

2.4 Literature review

The proposed methodology can be viewed from several angles. Here we focus on the linear kernel smoothing as well as the predictive neural network decomposition point of view.

First, we want to highlight that Nadaraya-Watson smoothers (Nadaraya, 1964; Watson, 1964) are usually for real-valued observations y with low-dimensional covariates $\mathbf{x}$. Under the assumption that the conditional expectation $\mathbb{E}(y \mid \mathbf{x})$ is a smooth function one can estimate by smoothing over the $\mathcal{X}$-space. This works well in low dimensions, but practical performance and also theoretical guarantees quickly deteriorate even for more than a couple of features (see also Geenens (2011) for an overview of "curse of dimensionality" in non-parametric regression). The proposed PRESS smoothers can be applied directly to binary and multi-label data, and also do not suffer from curse of dimensionality as demonstrated on the MNIST dataset with $p = 784$ features and $N = 55,000$ observations. For classification, a linear smoother does not have the direct interpretation of estimating a smooth function as ground truth, but rather one estimates the probability p. It may be for that reason that linear matrix smoothers are usually applied for regression problems only.

The neural network implementation of PRESS evolves naturally from statistical optimality in a predictive modeling framework. Cao et al. (2015) uses similar techniques for topic modeling in text data. Recently, interpretation and inference on neural networks has gained attention (Lipton, 2016; Lundberg and Lee, 2017). In particular, the information-theoretic explanation and analysis by Shwartz-Ziv and Tishby (2017) resembles the inherent optimality conditions and derivations of predictive states, in the information-theoretic sense of not losing information to predict. However, rather than a post-analysis of a given neural network, PRESS has interpretation and a fully probabilistic inference approach *built in*. Hence the two sides of interpretability as discussed in Lipton (2016), transparency and post-hoc explanation, are given by construction of how predictive states are defined in the first place.

We thus think that the PRESS framework provides a natural way to build transparent, interpretable, yet highly scalable and well-performing models.

3 Estimation

For classification $\mathbb{E}(y \mid \epsilon(\mathbf{x})) = p_{\epsilon(\mathbf{x})} = P(y \mid \epsilon(\mathbf{x}))$ and thus a PRESS classifier returns to its roots in the causal state literature (e.g., Shalizi, 2003; Shalizi and Crutchfield, 2001) in estimating optimal predictive *distributions*, rather than only expectations.

The latent-level model formulation in (11) suggests two ways to estimate $\mathbf{p}_S$: a) follow PRESS regression and use the weighted MLE in each state (Eq. (16)); or b) estimate $\mathbf{p}_S$ directly as part of the frequentist or Bayesian inference algorithm.[3]

Recall that we are not only interested in a good binary prediction of y_i, but want to obtain probabilistic predictions. Using the kernel matrix factorization in (17) the log-likelihood becomes

$$\ell(\Xi; \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{N} y_i \cdot \log\left(p(\mathbf{x}_i)\right) + \tag{19}$$
$$(1 - y_i) \cdot \log\left(1 - p(\mathbf{x}_i)\right)$$
$$= \sum_{i=1}^{N} y_i \cdot \log\left(\mathbf{w}_i(\theta)^\intercal \mathbf{p}_\mathbf{S}\right) + \tag{20}$$
$$(1 - y_i) \cdot \log\left(1 - \mathbf{w}_i(\theta)^\intercal \mathbf{p}_\mathbf{S}\right)$$

where $\Xi = (\theta, p_\mathbf{S})$ parametrizes the ϵ-mapping (θ) and – optionally – free parameters from the state-conditional Bernoulli distributions ($p_\mathbf{S}$). The maximum likelihood estimator (MLE)

$$\hat{\Xi} = \arg\max_{\Xi} \ell(\Xi; \mathbf{y}, \mathbf{X}), \tag{21}$$

can be obtained via numerical optimization.

[3] If $\mathbf{p}_S$ ($= \mu$) is considered a free parameter then the resulting model is not anymore equivalent to the symmetric factorization in (18), but a general matrix factorization, $\mathbf{S} = \mathbf{W}\mathbf{V}^\intercal$.

3.1 Bayesian inference

As PRESS is a fully probabilistic model, it naturally lends itself to Bayesian posterior sampling for inference and prediction – which can be difficult for alternative methods like random forests or SVMs. In particular, a PRESS model with linear ϵ mapping has a straightforward implementation in STAN (Carpenter et al., 2017).

Identifiability of state labeling: The state label assignment is arbitrary and a PRESS model is unidentifiable with respect to permutation of states. In order to maintain a consistent baseline state identification we follow the suggestion in Goerg (2017) and re-order state labels according to the (estimated) conditional mean, i.e., re-order states such that $0 \leq \hat{p}_{s_1} < \ldots < \hat{p}_{s_J} \leq 1$.

A disadvantage of this re-ordering is that states have to be estimated *before* imposing the order. This is not an issue for point estimates as they can just be re-labeled after estimation; for Bayesian inference this can become a problem as a) parallel chains are not necessarily in the same state ordering, and b) successive samples might jump back and forth between two (or more) equivalent PRESS representations, which wrongly suggest the chain has not yet reached an equilibrium.

We thus propose a reparameterization of the state-conditional probabilities $p_\mathbf{S}$ following a stick-breaking construction (Whye Teh et al., 2007). Let $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_J) \in (0, 1)^J$ and construct the state probabilities as

$$\begin{aligned} &p_{s_1} \leftarrow \rho_1, \\ &p_{s_j} \leftarrow p_{s_{j-1}} + (1 - p_{s_{j-1}}) \cdot \rho_j, j = 2, \ldots, J. \end{aligned} \tag{22}$$

This guarantees that $p_{s_{j-1}} < p_{s_j}$ for any $\boldsymbol{\rho}$.

3.2 Feature selection and feature importance

For feature selection we use an elastic net penalty (Zou and Hastie, 2005) on the coefficients $\boldsymbol{\theta}$ in the ϵ mapping. To get consistent results across different values of J the penalty per state is weighted by the probability of being in state j. That is we add a weighted elastic net penalty to the loss function,

$$R(\boldsymbol{\theta}) = \sum_{j=1}^{J} P(s_j) \cdot \left(\lambda_1 \sum_{k=1}^{p} |\theta_{k,j}| + \lambda_2 \sum_{k=1}^{p} \theta_{j,k}^2 \right). \tag{23}$$

Penalty parameters λ_1 and λ_2 can be chosen by cross-validation.

Recall that for identifiability, $\boldsymbol{\theta}$ is parametrized such that summing over j (states) adds up to 0 for each k (features). Hence we can view the weighted ℓ_1 norm as a measure of importance of feature k,

$$\iota(k) = \sum_{j=1}^{J} P(s_j) \cdot \|\theta_k\|_1 . \tag{24}$$

We use this measure to sort features by their importance in Figures 2, 5a and 6b.

4 Applications

We illustrate the presented methodology on several benchmark datasets and compare its predictive performance to alternative methods. We also show how to use PRESS for statistical inference and gain domain-specific insights.

Algorithms were implemented in TensorFlow, R, and STAN (Abadi et al., 2015; R Core Team, 2017; Carpenter et al., 2017).

4.1 Thoracic surgery

Here we illustrate the motivating example in Section 2 using the thoracic surgery dataset,[4] which contains data on $N = 376$ patients undergoing a lung cancer surgery and the risk in their post-operative life expectancy: $y = 1$ if patient dies within one year, 0 otherwise. Features to predict the risk are based on their health status such as their age or tumor size amongst others (see Fig. 2 for all features).

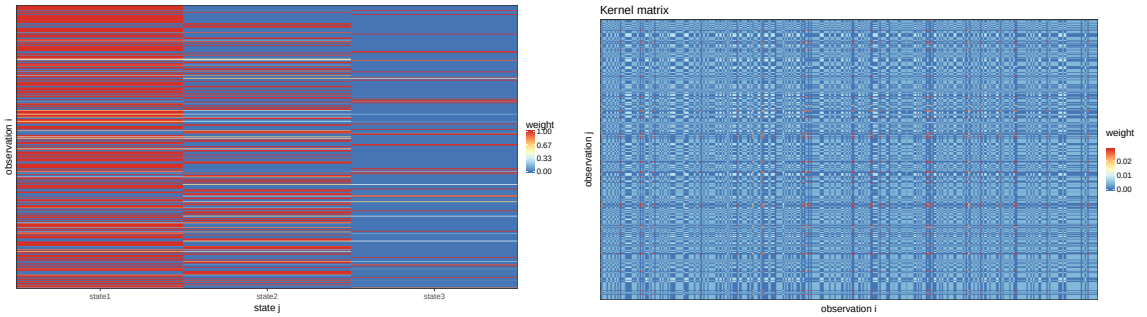
This is not only an important prediction problem, but also highlights the need for domain-specific inference, as for doctors as well as patients it is clearly important to know what contributes to a higher risk. The estimates of $\mathcal{S}$ and ϵ in Figure 1 and 2 tell us that – not surprisingly – tumor size and smoking contribute to higher risk (state 3) of complications, as the probability of death within one year in state 3 lies at 57%, whereas a "low" risk patient has an estimated 3% chance of dying (state 1).

For comparison, Figure 4 shows the estimated weights and kernel matrix for the $J = 3$ estimates with a LASSO penalization of $\lambda_1 = 0.02$ (as determined by cross-validation).

4.2 Iris dataset

The presented estimation and prediction methodology is not restricted to binary classification problems. As a simple example consider the iris dataset, which contains 3 species of flowers, "virginica", "setosa", and "versicolor". After transforming the labels to a 3-dimensional one-hot encoding, PRESS can estimate the state space and conditional predictive distributions for predicting

[4] Source: `https://archive.ics.uci.edu/ml/datasets/Thoracic+Surgery+Data`



(a) Probabilistic predictive state space $\mathbf{W} \in \mathbb{R}^{N \times 6}$. **(b)** Estimated kernel smoothing matrix $\mathbf{S} = \mathbf{S}(\mathbf{W})$.

Figure 4: Surgery dataset: predictive state estimation and metric learning results for $J = 3$ states

the species using same algorithms and formulas above on the multi-dimensional one-hot y.

A $J = 3$ model achieves a test data accuracy (80/20 split) of 93.3% accuracy; random forest predictions have 93.33% as well, but results are harder to interpret. As for the binary case PRESS reveals the state space, predictive distributions, and how covariates contribute to predictions in a straightforward interpretable way. Figure 5 summarizes the estimates.

4.3 Titanic

As yet another example consider the titanic dataset to predict whether a passenger has survived or not.[5].

Figure 6 shows ROC comparisons to Random Forest and generalized linear models for 20% test data, as well as weight estimates for the coefficients. In this example, the three models have (practically) the same performance, with PRESS being *interpretable* non-linear alternative.

4.4 MNIST

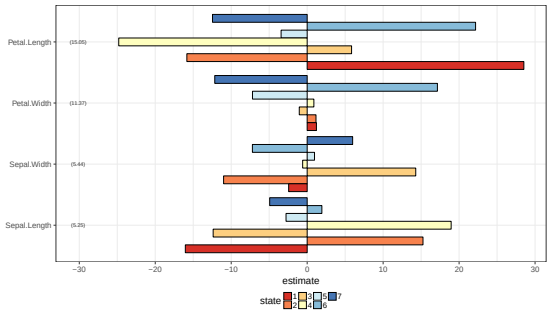
The handwritten digit dataset[6] is a good example to demonstrate the scalability of this kernel smoother. The data consists of $N_{train} = 55,000$ gray-scale images (28×28 pixels) of handwritten digits and their true value, $y \in \{0, \ldots, 9\}$. The gray-scale images are represented as $p = 28 \cdot 28 = 784$-dimensional feature vectors $\mathbf{x}_i \in \mathbb{R}^{784}$.

Naïvely a Nadaraya-Watson smoother would require a 784-dimensional kernel function $\mathcal{K}_{\mathbf{h}}(\|\mathbf{x} - \mathbf{x}_i\|)$ (e.g., Gaussian) with at least 784 bandwidth parameters h_k, $k = 1, \ldots, 784$. That kernel function is then applied to all pairwise $(\mathbf{x}_i, \mathbf{x}_j)$ and renormalized to represent an $N \times N$ smoothing matrix $\mathbf{S}$. For this dataset $\mathbf{S}$ has about $\mathcal{O}(N^2) \sim 3 \cdot 10^9$ entries. Even accounting for symmetry, storing this matrix requires writing and reading several GBs of memory, for just one iteration of an estimation algorithm.

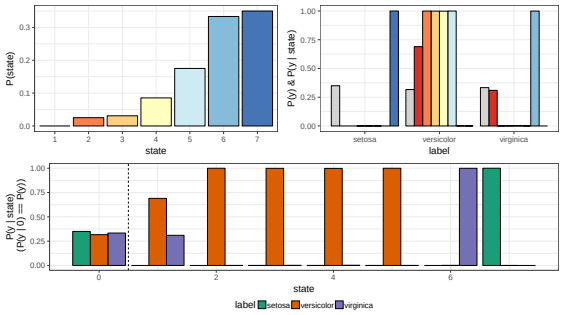
As discussed above, PRESS does not ever have to compute the full $N \times N$ matrix, but prediction and estimation scales linear in N, with the predictive state matrix $\mathbf{W} \in R^{N \times J}$ being the largest object that needs to be computed and stored $((J - 1) \cdot$

[5]Source: `http://biostat.mc.vanderbilt.edu/wiki/pub/Main/DataSets/`

[6]For reproducibility we use publicly available data in TensorFlow: `tf.contrib.learn.datasets.load_dataset('mnist')`.

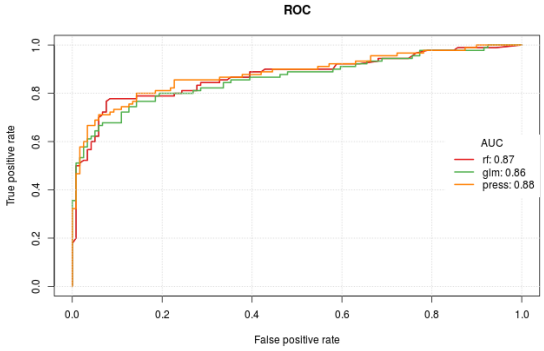


(a) Statistical inference to predict flower species.

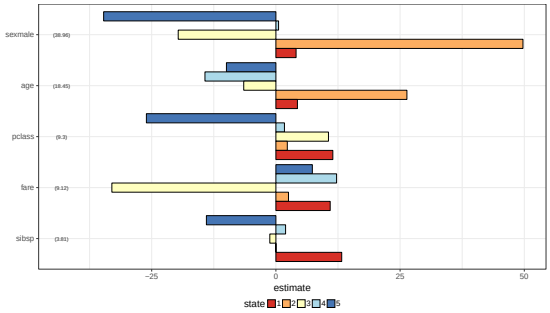


(b) Marginal and conditional distributions for states and species

Figure 5: Iris dataset: estimated $J = 7$ model with conditional distributions as well as coefficient estimates.



(a) Test data ROC and AUC estimate for PRESS , random forest, and multinomial regression.



(b) Estimated coefficients for $p = 5$ features to $J = 5$ states ordered by importance contributing to survival on Titanic. Higher states correspond to larger survival probability.

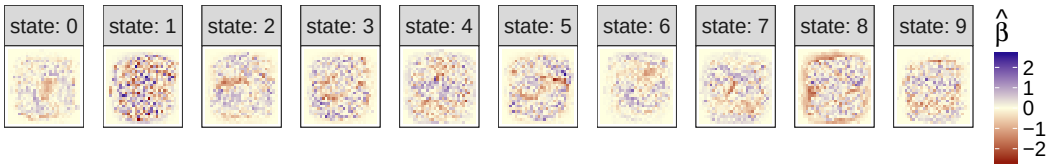
Figure 6: Titanic survival: model comparison with $J = 6$ states

$55,000 = 495,000$ free values; about 4 MB).

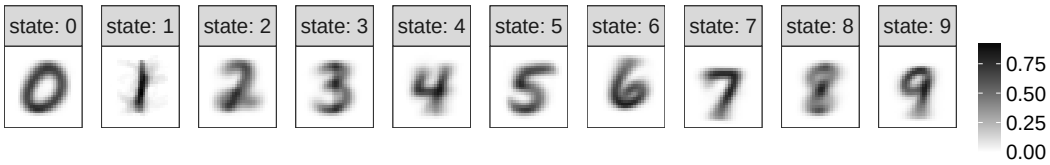
Figure 7a shows the estimated parametrization of the linear ϵ mapping to $J = 10$ predictive states. Non-zero coefficients highlight areas that are important to distinguish between states. They resemble patterns of associated true digit as well as digits that are similar in shape but have different ground truth.[7] For lack of displaying a 10 dimensional simplex, one way to interpret predictive states and their characteristics is to consider properties of the conditional distributions of the feature space given the states. In particular, Figure 7b shows (estimated) expected features given the states, i.e., $\mathbb{E}(\mathbf{X} \mid s_j)$. As expected the predictive states correctly identify the true digits. Figure 8a confirms this as the conditional distributions of each state are highly concentrated around one digit.

Another interesting quantity is the entropy of the ϵ mapping, i.e., $\mathcal{H}\left(p(S \mid \mathbf{x})\right) = -\sum_{j=1}^{J} p(s_j \mid \mathbf{x}) \log p(S \mid \mathbf{x})$, as it identifies those images that are easy (difficult) to classify. Figure 8 shows two

[7] We only use a linear mapping from features to states as it already gives meaningful and high quality results. One could add more layers to ϵ to achieve higher predictive accuracy, however, then loses the immediate interpretation of the weight estimates.

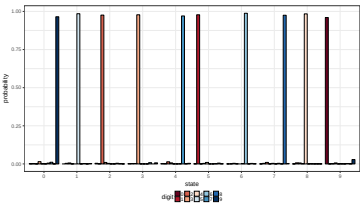


(a) Estimated coefficients for mapping $p = 784$ features to $J = 10$ states, $\hat{\epsilon} : \mathbb{R}^{784} \to \mathcal{S}$, $\mathbf{x}_i \mapsto s_j$, $j = 0, \ldots, 9$.

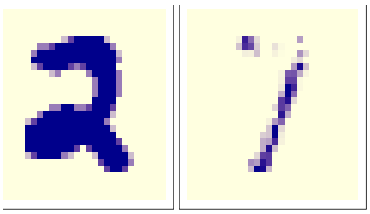


(b) State-conditional feature space represented as $\mathbb{E}(\mathbf{X} \mid S)$.

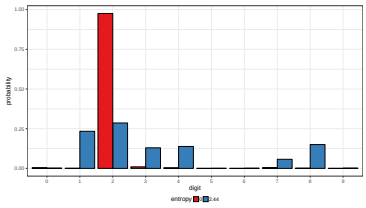
Figure 7: PRESS estimates for the MNIST dataset with $J = 10$ states.



(a) State-conditional distributions $p(y \mid s_j)$, $j = 1, \ldots, 10$ with $K = 10$ digit labels per state.



(b) Observations with lowest (left) and highest (right) entropy of their probabilistic state distribution $p(s_j \mid \mathbf{x}_i)$ among test data.



(c) Conditional distribution estimate $p(s_j \mid \mathbf{x}_i)$, $i \in \{low, high\}$ for the low and high entropy observations displayed in 8b.

Figure 8: PRESS estimates for the MNIST dataset with $J = 10$ states.

examples from the test data with lowest and highest entropy, respectively. While the left image is clearly a 2 (entropy practically 0), the right image is mapped to various states (higher entropy).

5 Summary & Discussion

We introduce *predictive state smoothing (PRESS)* classification, a semi-parametric kernel classifier for high-dimensional data and binary or multi-label response. PRESS is a metric learner, which determines that kernel function which gives the best probabilistic predictions for y given $\mathbf{x}$. It is not only statistically optimal in a theoretical sense, but also computationally efficient as prediction and estimation can rely on the kernel trick and thus compute predicted values linearly in N (instead of $O(N^2)$ for standard non-parametric kernel smoothing methods). We also embed PRESS in a deep neural network framework with specific layer and activation functions that conforms to the probabilistic basis of predictive states. It scales well in the number of variables and allows for LASSO or Ridge like variable selection for the $p \gg N$ case. We present algorithms for maximum likelihood estimation as well as Bayesian inference, which can be easily implemented in TensorFlow, R, or STAN. PRESS compares well with state-of-the-art classification

techniques, yet it remains interpretable and can be used for statistical inference to obtain domain-specific insights as shown on several real world datasets.

Acknowledgments

I want to thank Joseph Kelly, Jim Koehler, and Jing Kong for valuable discussions and feedback on this work; Ashish Saxena, Xiaojing Wang, and Yunting Sun for suggestions to improve the implementation of PRESS in TensorFlow. I also want to thank Johannes Singer for expert advice on the thoracic surgery data.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org.

Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32.

Cao, Z., Li, S., Liu, Y., Li, W., and Ji, H. (2015). A novel neural topic model and its supervised extension. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 2210–2216. AAAI Press.

Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software, Articles*, 76(1):1–32.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Dawid, A. P. (1979). Conditional Independence in Statistical Theory. *Journal of the Royal Statistical Society. Series B (Methodological)*, 41(1):1–31.

Geenens, G. (2011). Curse of dimensionality and related issues in nonparametric functional regression. *Statist. Surv.*, 5:30–43.

Goerg, G. M. (2017). Predictive State Smoothing (PRESS): A scalable non-parametric regression for high-dimensional data with variable selection. Technical report, Google. Submitted for publication.

Hearst, M. A. (1998). Support vector machines. *IEEE Intelligent Systems*, 13(4):18–28.

Hofmann, T., Schölkopf, B., and Smola, A. J. (2008). Kernel methods in machine learning. *The Annals of Statistics*, 36(3):1171–1220.

Lauritzen, S. L. (1974). On the Interrelationships Among Sufficiency, Total Sufficiency and Some Related Concepts. Technical report, Stanford University, Department of Statistics.

Lipton, Z. C. (2016). The mythos of model interpretability. *CoRR*, abs/1606.03490.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems 30*, pages 4768–4777. Curran Associates, Inc.

Nadaraya, E. A. (1964). On Estimating Regression. *Theory of Probability and Its Application*, 9:141–142.

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117. cited By 1014.

Shalizi, C. R. (2003). Optimal Nonlinear Prediction of Random Fields on Networks. In Morvan, M. and Rémila, É., editors, *Discrete Models for Complex Systems, DMCS'03*, volume AB of *DMTCS Proceedings*, pages 11–30. Discrete Mathematics and Theoretical Computer Science.

Shalizi, C. R. and Crutchfield, J. P. (2001). Computational mechanics: Pattern and prediction, structure and simplicity. *Journal of statistical physics*, 104(3):817–879.

Shwartz-Ziv, R. and Tishby, N. (2017). Opening the black box of deep neural networks via information. *CoRR*, abs/1703.00810.

Watson, G. S. (1964). Smooth regression analysis. *Sankhy: The Indian Journal of Statistics, Series A (1961-2002)*, 26(4):359–372.

Whye Teh, Y., Görür, D., and Ghahramani, Z. (2007). Stick-breaking Construction for the Indian Buffet Process. 2:556–563.

Zou, H. and Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320.

