Classification with Features Missing at Test Time

Sergey Karayev

Abstract

Problem

- Classification: all features are available at training time, but only a subset is observed at test time for each instance.
- Arises in situations with sensor loss, or when using test-time instance-specific feature selection (our motivation).

Approach

- We compare several ways of completing missing variables:
  - **mean** imputation; **SVD** imputation; conditioning a **joint Gaussian**; and averaging **k-Nearest Neighbors**.
- For classification, we compare:
  - logistic regression trained on **fully observed** data, and on **synthetically imputed** data; using **marginalized corrupted features** loss function; and **k-Nearest Neighbors**.

Evaluation

- **Digits**: features are pixel values;
- **Scenes**: features are SVM outputs from different kernels.

Problem Formulation

Classification Problem

- N instances of K possible labels: $\mathcal{D} = \{x_n \in \mathcal{X}, y_n \in \{1, \dots, K\}\}_{n=1}^{N}$.
- Set $\mathcal{H}$ of F features $\mathcal{H} = \{h_f : \mathcal{X} \mapsto \mathbb{R}\}_{f=1}^{F}$.
  To represent featurized instance, we write $\mathbf{x} = [h_0(x), h_1(x), \dots, h_f(x)]$.
- Test-time feature selection function $o(x, b) : \mathcal{X} \times \mathbb{R} \mapsto \mathbb{B}^F$, where $\mathbb{B} \in \{0, 1\}$, and $b \in [0, 1]$ specifies the fractional selection *budget*.
- Applied to x, $o(x, b)$ gives the binary selection vector $\mathbf{o}$ which splits $\mathbf{x}$ it into observed and unobserved parts such that $\mathbf{x}^m = [\mathbf{x}^\mathbf{o}, \mathbf{x}^\mathbf{u}]$.
- X^c denotes the fully observed $N \times F$ training set. We assume that we have only sequential access to the missing-feature test instances $\mathbf{x}^m$.

For each b, we seek a procedure $g(x)$ that minimizes total loss $\sum \ell(g(x_n), y_n)$ under the condition that $o(x, b)$ is applied to each x.

Futher Work

Distribution of $o(x, b)$

Currently we consider the feature selection function to be random given the budget, but the goal is to consider non-factorized distributions.

References

[1] L. Van Der Maaten, M. Chen, S. Tyree, and K. Q. Weinberger, "Learning with Marginalized Corrupted Features," in ICML, 2013, vol. 28.

Missing Value Imputation

Since our training data is fully observed, we can train classifiers to rely on all features. At test time, when not all features are observed, the first reasonable thing is to impute their values.

Mean imputation

We simply replace a missing value $\mathbf{x}_f$ with its mean in X^c_f, which is 0.

SVD imputation

The rank-R truncated SVD of $N \times F$ matrix X^c can be written as

$$\hat{X}^c = U_R D_R V_R^T \tag{1}$$

We know that this is the best rank-R approximation to X^c, which means that SVD also gives the least squares solution to regression of $\mathbf{x}^c$ onto the right singular vectors.

The same regression can be performed using only observed values:

$$\min_\theta \sum_{f\text{obs}} (\mathbf{x}^\mathbf{m}{}_f - \sum_{r=1}^{R} v_{fr}\theta_r)^2 = \min_\theta \left\| \mathbf{x}^m - V_R^\mathbf{o}\theta \right\| \tag{2}$$

where V_R^o is a version of V_R composed only of rows corresponding to observed features in $\mathbf{x}^m$.

The unobserved values $\mathbf{x}^\mathbf{u}$ are filled in by $V_R^\mathbf{u} \left(V_R^{\mathbf{o}T} V_R^o\right)^{-1} V_R^{\mathbf{o}T} \mathbf{x}^\mathbf{o}$.

Parameters: R is set by cross-validation on the training set.

Gaussian imputation

In this model, we assume that $\mathbf{x}$ is distributed according to a multivariate Gaussian $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, where Σ is the sample covariance $X^{cT}X^c$.

In case of partially observed features, we can write

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}^\mathbf{o} \\ \mathbf{x}^\mathbf{u} \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \mathbf{A} & \mathbf{C} \\ \mathbf{C}^T & \mathbf{B} \end{bmatrix}\right) \tag{3}$$

where C is the cross-variance matrix that has as many rows as the size of $\mathbf{x}^\mathbf{o}$ and as many columns as the size of $\mathbf{x}^\mathbf{u}$.

The distribution over unobserved variables conditioned on the observed variables is simply

$$\mathbf{x}^\mathbf{u} \mid \mathbf{x}^\mathbf{o} \sim \mathcal{N}\left(\mathbf{C}^T\mathbf{A}^{-1}\mathbf{x}^\mathbf{o},\ \mathbf{B} - \mathbf{C}^T\mathbf{A}^{-1}\mathbf{C}\right) \tag{4}$$

k-NN imputation and classification

Instead of relying on information contained in the covariance matrix, we could go directly to the data to impute missing values.

For $\mathbf{X}^m$, we find the k nearest neighbors with the highest dot product similarity $\mathbf{x}^{cT}\mathbf{x}^\mathbf{m}$ or lowest Euclidean distance $\|\mathbf{x}^c - \mathbf{x}^m\|$, using only the features that are observed in $\mathbf{x}^m$.

For **imputation**, the unobserved values are set to the average across these nearest neighbors for that dimension.

Similarly, we do **classification** by returning the mode label of the nearest neighbors.

Parameters: k is set by cross-validation on the training set, independently for imputation and classification.

Classifier

In addition to the k-NN classifier, we consider three linear classifier, obtained by minimizing:

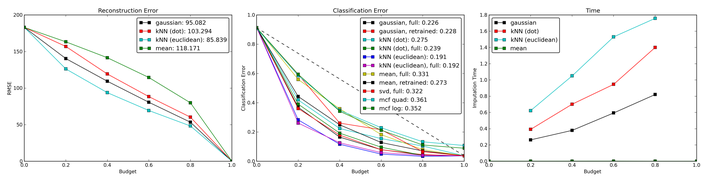
1. Logistic loss on fully observed data: simply using X^c.
2. Logistic loss on synthetically re-imputed version of the data: we obtain X^m through applying $o(x, b)$ on each x^c, and then imputing the missing values with the methods described.
3. Logistic and quadratic loss derived with Marginalized Corrupted Features (MCF) [1].

The idea of (3) is to work the expectation of feature corruption into the loss function, which is functionally equivalent to adding an infinite amount of corrupted data to the training set, but remarkably is not any more computationally complex than the standard loss.

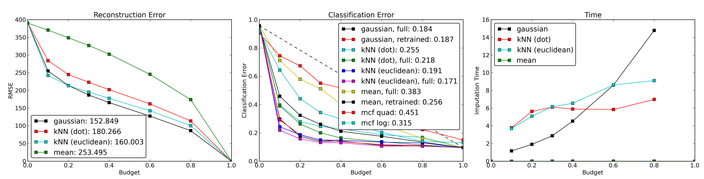
For example, for blankout noise and logistic loss, we can derive the surrogate loss

$$\mathcal{L}(D; \theta) \leq \sum_{n=1}^{N} \log\left(1 + \prod_{f=1}^{F}\left[q_f + (1 - q_f)e^{-y_n\theta_f \frac{1}{1-q_d} x_{nf}}\right]\right) \tag{5}$$

Evaluation



Random policy on the **Digits** dataset: 10 classes with 200 samples per class, feature dimension 64 (pixels).



Random policy on the **Scenes-15** dataset: 15 classes, feature dimension is 165 (SVM outputs of 11 different feature kernels).

- The MCF loss classifier is outperformed by all approaches, including simple mean imputation with retraining.
- Retraining the classifier with synthetically imputed data improves performance, but much more dramatically for simple approaches than for already well-performing imputations.
- The most expensive method—k-NN—performs the best.