Automatic classification of variables in catalogs with missing data

Pavlos Protopapas

Institute for Applied Computational Science
Harvard-Smithsonian Center for Astrophysics

January 29, 2013

Outline

- Problem
- Bayesian Networks
- Inferences using Bayesian Networks
- Impute data
- Classification
- Experiments

Problem

- Classifying objects based on their features (color and magnitude) dates back in the 19th century.
- Automatic classification
  - Recently became sophisticated
  - Necessary due to the exponential growth of astronomical data.
- In time-domain astronomy $\rightarrow$ of light-curves
  - Use features of the light-curves
  - Apply sophisticated machine learning to classify objects in a multidimensional features space, provided there are enough examples to learn from (training).
  - After almost a decade since the first appearance of automatic classification methods, many of those methods have produced and continue to produce high fidelity catalogs

Kim et al 2011,2012, Bloom 2011, Richards 2011, Debosscher 2007, Wachman 2009, Wang 2010

Problem

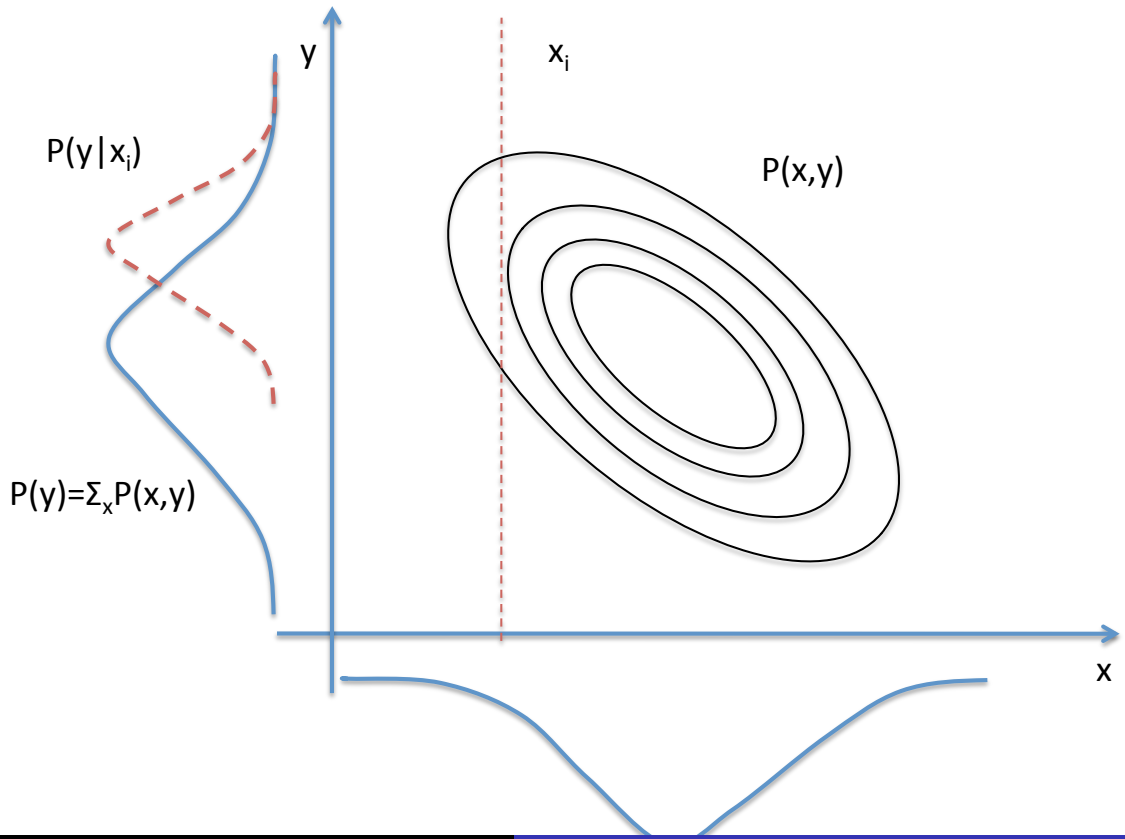
Best to use as many available catalogs as possible (x-ray, u-band, time series catalogs)

- Catalogs are taken with different instruments, bandwidths, locations, times, etc,
- The intersection of these catalogs is smaller than any single catalog $\rightarrow$ resulting multi-catalog contains missing values.

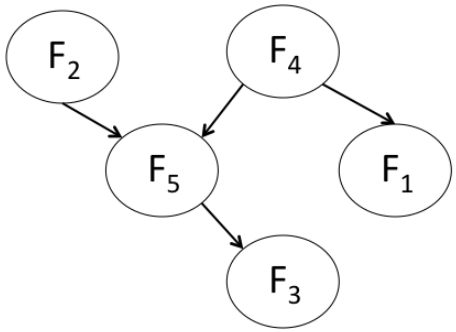
Traditional classification methods can not deal with the resulting catalogs

Introduction

- Fill missing data using Monte Carlo approaches. Each missing value is drawn from a distribution that is determined from all objects in the training set.
- This approach totally ignores the relationship amongst the variables.



Bayesian Networks (BN)



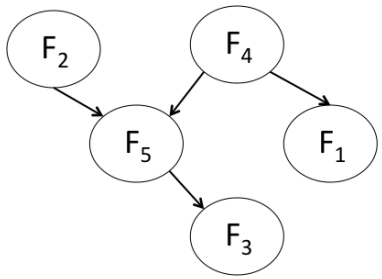
A Bayesian network (BN) is a probabilistic graphical model that represents dependency relationships among a set of variables.
One of the main advantages of the BN factorization is that each of the factors involves a smaller number of variables, where it is easier to estimate.

Bayesian Networks (BN)

[t] Let $S = \{x_1, \ldots, x_n\}$ be a set of data instances, each one described with a set of D variables $\{F_1, \ldots, F_D\}$. Each instance x_i is represented as a vector $x_i = \{F_1^i, \ldots, F_D^i\}$.
BNs can represent the joint probability distribution $P(F_1, \ldots, F_D)$ of dataset S as a product of factors, where each factor is a conditional probability distribution of each node given its parents in the BN:

$$P(S) = \prod_{i=1}^{n} P(x_i) = \prod_{i=1}^{n} P(F_1^i, \ldots, F_D^i) = \prod_{i=1}^{n} \prod_{j=1}^{D} P(F_j^i | Pa_{BN}^i(F_j))$$

where $\mathrm{Pa}_{BN}(F_j)$ represents the set of parents of variable F_j in the BN and $\mathrm{Pa}_{BN}^i(F_j)$ indicate that parents of feature F_j are instantiated in the values of x_i



joint probability distribution can be factorized according to the BN as:

$$P(F_1, \ldots, F_5) = P(F_1|F_4)P(F_2)P(F_3|F_5)P(F_4)P(F_5|F_2, F_4)$$

Inferences

- Suppose we have an object with missing values F_5 and F_2.
- Estimate the most probable value for variable F_5 given the observed values for F_1, F_3, F_4, we can calculate $P(F_5|F_1, F_3, F_4)$ as:

Factorizing

$$P(F_5|F_1,F_3,F_4) = \frac{P(F_1,F_3,F_4,F_5)}{P(F_1,F_3,F_4)}$$

$$= \frac{\sum_{F_2} P(F_1,F_2,F_3,F_4,F_5)}{\sum_{F_2,F_5} P(F_1,F_2,F_3,F_4,F_5)}$$

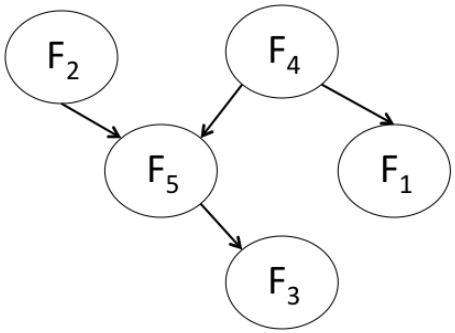
$$= \frac{\sum_{F_2} P(F_1|F_4)P(F_2)P(F_3|F_5)P(F_4)P(F_5|F_2,F_4)}{\sum_{F_2,F_5} P(F_1|F_4)P(F_2)P(F_3|F_5)P(F_4)P(F_5|F_2,F_4)}$$

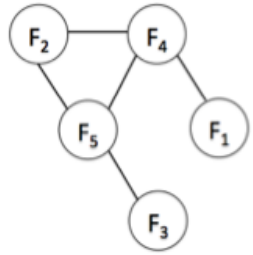
$$= \frac{P(F_1|F_4)P(F_3|F_5)P(F_4)\sum_{F_2} P(F_2)P(F_5|F_2,F_4)}{P(F_4)P(F_1|F_4)\sum_{F_2,F_5} P(F_3|F_5)P(F_2)P(F_5|F_2,F_4)}$$

Pusing in

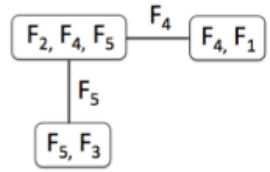
- "Pushing in" the factors in the sums is known as *variable elimination* (Pearl et el 94)
- The simplest exact inference algorithm.

Juntion Tree





Moral Graph

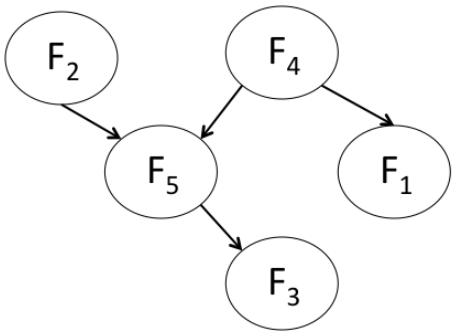


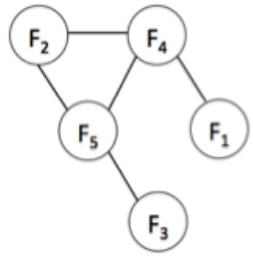
Junction Tree

A junction tree is an undirected graph where each node corresponds to a set of variables in the original BN that forms a family (a node with its parents).

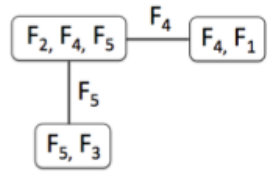
- The moral graph is such that two nodes are connected if either they are directly connected in the BN or if they have a common child in the BN.
- Identify all the cliques (sets of nodes where every pair inside the set is connected by an edge) in the moral graph. The junction tree is a new data structure which has one node per each clique in the moral graph, and two nodes are connected if they have at least one common feature between them.

Juntion Tree





Moral Graph

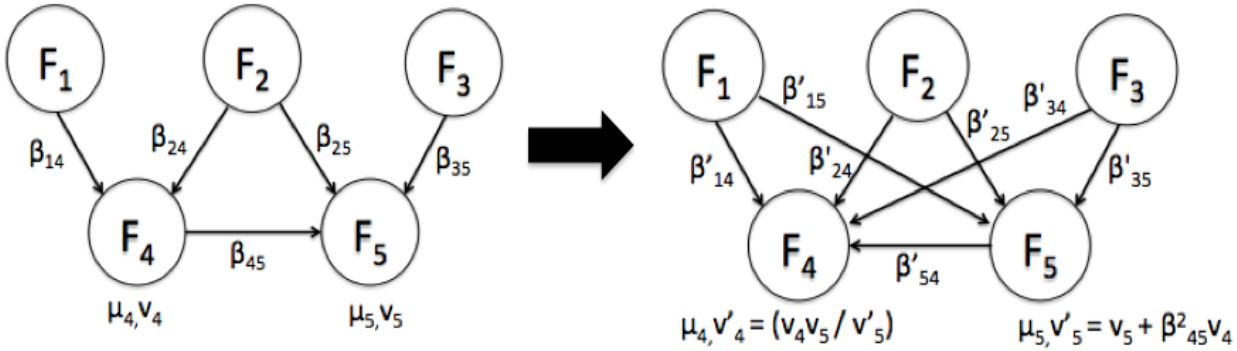


Junction Tree

- Nodes send to other nodes the value of its own probability, summing out the variables which are not included in the destination node.
  - $\{F_2, F_4, F_5\}$ sends $\sum_{F_2} \sum_{F_5} P(F_5|F_2, F_4)$ to $\{F_4, F_1\}$
  - $\{F_2, F_4, F_5\}$ sends $\sum_{F_2} \sum_{F_4} P(F_5|F_2, F_4)$ to $\{F_5, F_3\}$
  - $\{F_5, F_3\}$ sends $\sum_{F_3} P(F_3|F_5)$ to $\{F_2, F_4, F_5\}$
  - $\{F_4, F_1\}$ sends $\sum_{F_1} P(F_1|F_4)$ to $\{F_2, F_4, F_5\}$

  This process is known as 'junction tree calibration" (Cowell 2002).

Gaussian Nodes inference



$$\beta'_{14} = \beta_{14} - \beta'_{15}\beta'_{54} \qquad \beta'_{54} = \beta_{45}\ (v_4 / v'_5) \qquad \beta'_{35} = \beta_{35}$$

$$\beta'_{15} = \beta_{14}\beta_{45} \qquad \beta'_{25} = \beta_{25} + \beta_{24}\beta_{45}$$

$$\beta'_{24} = \beta_{24} - \beta'_{25}\beta'_{54} \qquad \beta'_{34} = -\beta'_{35}\beta'_{54}$$

- Continuous data $\rightarrow$ Gaussian Nodes inference (Shachter 1989).
  Variables not involved in the calculation of a probability can be eliminated from the Bayesian network (barren nodes). e.g. leaf nodes
- Not so simple. Arc reversal

Learning Bayesian Networks with complete data

- Learning the network involves learning the structure (edges) and the parameters (probability distributions on each of the factors).

Structure Learning with complete data

- Number of possible networks structure grows exponentially $\rightarrow$ a greedy search strategy.
- Starting with an initial random order of variables (nodes), we add parents to the nodes incrementally and keep the parent(s) that generates the network with the highest score.
- The score of a network structure is related to how the structure fits data.
  Probability of the structure given the data, which corresponds to apply the same factorization imposed by the structure and use multinomial distributions over each factor. We estimate each probability by firstly discretizing the possible values that each feature F_i can take and then creating a multi-dimensional histogram.

Parameter Learning with complete data

Learning the parameters of a BN means to learn the distribution of each of the factors given by the structure of the BN.
$\Rightarrow$ To learn the parameters it is necessary first to know the structure.

Parameter estimation

Use Gaussians to model these probability distributions. We model F_j as a linear Gaussian of its parents:

$$P(F_j) = \mathcal{N}(\beta_0 + \beta^T \mu; \sigma^2 + \beta^T \Sigma \beta), \tag{1}$$

where the set of parents $Pa_{BN}(F_j)$ are jointly Gaussian $\mathcal{N}(\mu; \Sigma)$. Note that μ and β are k dimensional vectors, and the matrix Σ is $k \times k$

To learn a Gaussian node, we first learn (i) the Gaussian distribution of each of the parent nodes and then (ii) the set of parameters $\{\beta_0, \ldots, \beta_k; \sigma\}$ of the linear combination.

Learning Gaussians for each parent node

We model each parent node as a mixture of Gaussians.

- To estimate their distribution use Expectation Maximization (EM) algorithm.
- EM optimizes the likelihood function of a model which depends on latent or unobserved variables.

Learning the linear combination of parents

Log-likelihood

Having determined the distribution of each parent node, we can now estimate the distribution of each Gaussian node F_j given its parents. Let $\{U_1, \ldots, U_k\}$ be the parent nodes with respective means $\{\mu_1, \ldots, \mu_k\}$, then $P(F_j|Pa_{BN}(F_j)) = \mathcal{N}(\beta_0 + \beta_1\mu_1 + \cdots + \beta_k\mu_k; \sigma^2)$. Our task is to learn the set of parameters $\theta_{F_j} = \{\beta_0, \ldots, \beta_k; \sigma\}$. To learn those parameters we optimize the log-likelihood, expressed as:

$$\mathcal{L}_{F_j}(\theta_{F_j}|D) = \sum_{i=1}^{n} \left[-\frac{1}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}(\beta_0 + \beta_1\mu_1 + \beta_k\mu_k - x_i)^2 \right].$$

Setting the derivative with respect to $\beta_1, \ldots, \beta_k$ to zero we have the following k equations:

$$\begin{aligned} E[F_j \cdot U_1] &= \beta_0 E[U_1] + \beta_1 E[U_1 \cdot U_1] + \beta_k E[U_k \cdot U_1] \\ &\vdots \\ E[F_j \cdot U_k] &= \beta_0 E[U_k] + \beta_1 E[U_1 \cdot U_k] + \beta_k E[U_k \cdot U_k] \end{aligned}$$

$$\sigma^2 = \mathrm{cov}[F_j, F_j] - \sum_{p=1}^{k}\sum_{q=1}^{k} \beta_p\beta_q \,\mathrm{cov}[U_p, U_q]$$

Learning Bayesian Networks with missing data

How to learn both the structure and the parameters under incomplete data?

To learn the parameters with missing data, we need to previously know the structure and to learn the structure we need to guess the missing values.

$\rightarrow$ This is done in a iterative method.

Learning parameters with missing data

We assume that we already know the structure of the Bayesian network before we start learning the distribution parameters with missing data.

$\rightarrow$ Optimize the log likelihood given the data.

Likelihood

- Each data point x_i can be written as

$x_i = \{x_i^o, x_i^m\}$

Rewriting the log likelihood we have:

$$
\begin{aligned}
l(\theta|x^o, x^m, z) \quad = \quad & \sum_{i=1}^{n}\sum_{j=1}^{C} z_{ij}\Big[\frac{n}{2}\log 2\pi + \frac{1}{2}\log\sigma_j \\
& - \frac{1}{2\sigma_j^{2,o}}(x_i^o - \mu_j^o)^2 \\
& - \frac{1}{2\sigma_j^{2,om}}(x_i^o - \mu_j^o)(x_i^m - \mu_j^m) \\
& - \frac{1}{2\sigma_j^{2,m}}(x_i^m - \mu_j^m)^2\Big]
\end{aligned}
$$

Learning parameters with missing data

Likelihood

Note that in the E-step for the missing data case we need to estimate three unknown terms, z_{ij}, $z_{ij}x_i^m$, and $z_{ij}x_i^{m2}$. Then the E-Step computes:

$$\begin{aligned} E[x_i^m | z_{ij} = 1, x_i^o, \theta_k] &= \mu_j^m + \frac{(x_i^o - \mu_j^o)}{\sigma_j} \\ E[z_{ij}x_i^m | x_i^o, \theta] &= P(z_{ij}|x_i, \theta)x_{ij}^m \\ E[z_{ij}(x_i^m)^2 | x_i^o, \theta] &= P(z_{ij}|x_i, \theta)((x_{ij}^m)^2) \end{aligned}$$

The M-Step for the missing case is the same as in the complete data case, the main difference is just that the unknown values are replaced by the expectations in equations above.

Learning the network structure with missing data

To learn the structure of a Bayesian network with missing data, we complete the missing values and then iterate to improve these values using the structure learned so far.

Algorithm

- Learn for each variable in $\{F_1, \ldots, F_D\}$ a univariate Gaussian Mixture $GM(i)$ $i \in [1 \ldots D]$.
- Create M complete datasets D_s^1, $s \in [1 \ldots M]$ filling the missing values of each variable F_i with values sampled from $GM(i)$.
- $t = 1$

while Convergence criteria is not achieved **do**

  for $s = 1$ to M **do**

    - From each complete dataset in $D_s^{(t)}$, learn a BN, $B_s^{(t)}$.

  end for

  - Create one Bayesian network structure $B^{(t)}$ as the union of all the BNs.
  - Learn the parameters $\theta^{(t)}$ using the original incomplete data and the network structure $B^{(t)}$
  - Use the network $\langle B^{(t)}, \theta^{(t)} \rangle$ to sample new values and create new completed datasets $D_s^{(t+1)}$.
  - $t = t + 1$

end while

The automatic classification model

So far, infer the missing values. Next:

- Automatic classifier using the new training completed set.
- Random Forest (RF) classifier
  - Very efficient algorithm based on decision tree models and bagging for classification problems
  - Ensemble methods, appearing in machine learning literature at the end of nineties and has been used recently in the astronomical journals .
    Quinlan 1993, Breiman 2001, Pichara 2012

Random Forrest

The process of training or building a RF given training data is as follows:

- Let P be the number of trees in the forest and F the number of features in each tree, both values are model parameters.
- Build P sets of n samples taken with replacement from the training set. Note that each of the P bags has the same number of elements with the training set but less different examples, given that the samples are taken with replacement (The training set also has n samples).
- For each of the P sets, train a decision tree using a random sample of F features from the set of q possible features.

Random Forrest

RF creates many linear separators, attempting to separate between elements of different classes using some featuresand some data points (the ones given by each of the bags).

Each of the decision trees creates one decision
The final decision is the most voted class among the set of P decision trees

As the number of trees goes to infinity the classification error of the RF becomes bounded and the classifier does not overfit the data.

Experimental Results

To test the advantage of the model, we ask the following questions

- is it possible to learn an automatic classification model that is able to deal with missing data?
- does the model outperform any model that uses a subset of training set with complete data?
- does the proposed model overcome the case where missing data is filled using models that treat each variable independently?

Experimental Results

- Base catalog, the MACHO catalog and extract 14 features from each lightcurve.
- Combine the MACHO catalog with other catalogs containing magnitudes at different wavelengths.
  - SAGE (Meixner et al 2006)
  - UBVI (Piatti 2011)
  - 2MASS (Skrutskie 2006)

Data

Percentage of missing values on 2MASS/SAGE catalogs

Vars	% of missing values
J	54%
H	54%
K	58%
m_{36}	1%
m_{45}	13%
m_{58}	68%
m_{80}	74%

Percentage of missing values on UBVI catalog

Vars	% of missing values
U	49%
B	0%
V	0%
I	14%

Data

Number of objects per class on SAGE-2MASS-UBVI training set

Class	number of training objects
Non-Variables	1136
QSO	45
Be star	76
Cepheid	70
RR Lyrae	69
EB	100
LPV	337

Results

Class	Gaussian mixture Precision	Recall	F-Measure	Our model Precision	Recall	F-Measure
None Variables	0.857	0.952	0.902	**0.878**	0.942	**0.909**
Quasar	0.9	0.8	0.847	0.878	**0.956**	**0.915**
Be	0.679	0.5	0.576	**0.724**	**0.553**	**0.627**
Cepheid	0.805	0.886	0.844	0.785	0.886	0.832
RR-Lyrae	0.333	0.014	0.028	**0.583**	**0.203**	**0.301**
EB	0.5	0.25	0.333	**0.525**	**0.31**	**0.39**
LPV	0.919	0.938	0.928	**0.925**	**0.947**	**0.935**
Weighted Average:	0.821	0.851	0.826	**0.846**	**0.863**	**0.848**

Moreover, after training the model, we run it on the whole MACHO catalog, in order to generate a new quasar candidate list. To evaluate the quality of the new quasar candidate list, we calculate the matching level of our list of candidates with previous known lists.

	Adding SAGE-2MASS-UBVI features	Only MACHO Features
Quasar Precision	0.858	0.857
Quasar Recall	**0.956**	0.8
Quasar F-Score	**0.896**	0.828

Conclusions

- A new way of dealing with missing data
- Tested on real astronomical datasets
  - Catalogs with missing data can be useful for automatic classification.
  - Integrate catalogs improve performance
  - Improve accuracy of our results in previous work on quasar detection
- Considers probability dependencies between variables, that makes possible to take advantage of the observed values, in order to increase the accuracy of the estimation when the number of observed values increase.