Using Query-Specific Variance Estimates to combine Bayesian Classifiers

Chi-Hoon Lee, Russ Greiner, and Shaojun Wang
Department of Computing Science, University of Alberta, Edmonton, AB,Canada

Introduction

Ensemble Approach

to produce an effective classifier

- Learn many base-classifiers
  - -- Different bootstrap samples
  - -- Different learning algorithms
- To classify new instance ***e***:
  - -- Evaluate each base classifier
  - -- Combine using "**Combination Rule**"

$f_{all}(\mathbf{e}) = \Sigma_i\ \alpha_i\ f_i(\mathbf{e})$

Data–Independent weight

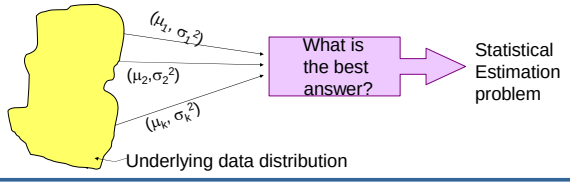
Propose an Ensemble -- MUV

- Different Weights for different instances
- Use **Variance** to approximate the weights
- **Combining Responses as a Statistical Estimation problem**

$f_{all}(\mathbf{e}) = \Sigma_i\ \alpha_i(\mathbf{e})\ f_i(\mathbf{e})$

Data–Dependent weight

P(cancer | gender=female,smoke=true,age=old)?



Foundations 1 – Combining responses

Combination Principle

Assume correct answer is λ
Only see (independent) estimates

$\mu_i = \lambda + \varepsilon_i$

where $\varepsilon_i \sim N(0, \sigma_i^2)$

Estimate λ, given $\{\langle \mu_i, \sigma_i^2 \rangle\}_i$

Maximum Likelihood (MLE)
--- Unbiased

$$\hat{\lambda}_{MLE} = \frac{\sum_{i=1}^{k} \frac{\mu_i}{\sigma_i^2}}{\sum_{i=1}^{k} \frac{1}{\sigma_i^2}}$$

James-Stein Shrinkage (JS)
--- Biased

$$\hat{\lambda}_{JS} = \left(1 - \frac{\sum_{i=1}^{k} \frac{1}{\sigma_i^2}}{\left(\sum_{i=1}^{k} \frac{\mu_i}{\sigma_i^2}\right)^2}\right)^+ \times \hat{\lambda}_{MLE}$$

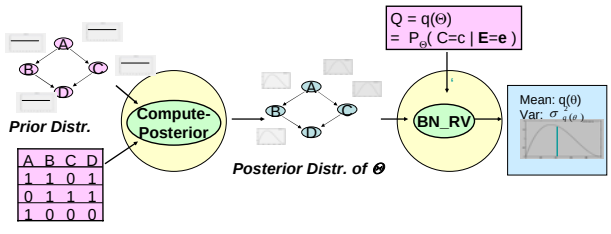
MUV – Mixture Using Variance

Proposed model to combine Bayesian Classifiers

- Given evidence E = *e*
- *i*th classifier computes for a response ***c***
  - Expectation: $m_i(c) = E[\ P_i(c|\mathbf{e})\ ]$
  - Variance : $v_i(c) = Var[\ P_i(c|\mathbf{e})\ ]$
- MUV query response (MLE)

$$c^* = \arg\max_{c \in C} \frac{1}{\sum_i \frac{1}{v_i(c)}} \sum_i \frac{m_i(c)}{v_i(c)}$$

Foundations 2a – Computing Belief Net Variances



- $E[P(c \mid e, \Theta)] = P(c \mid e, E[\Theta]) = P_{\bar{\Theta}}(c \mid e)$
- $\bar{\sigma}^2\ [P(c \mid e, \Theta)] = \sum_{\theta_{D|f} \in \Theta} \frac{1}{1 + n_{D|f}} \bar{v}_{c|e}(D \mid f)$

$$\bar{v}_{c|e}(D \mid f) = \sum_{d \in D} \frac{1}{\bar{\theta}_{d|f}} \left[P_{\bar{\Theta}}(d, f, c \mid e) - P_{\bar{\Theta}}(c \mid e) P_{\bar{\Theta}}(d, f \mid e)\right]^2 - \left[P_{\bar{\Theta}}(f, c \mid e) - P_{\bar{\Theta}}(c \mid e) P_{\bar{\Theta}}(f \mid e)\right]^2$$

- $\bar{\Theta}$: posterior means of Θ
- $\theta_{D|f} = \theta_{D|\alpha(D)=f} \sim Dir(\alpha_{D=1|f}, \ldots, \alpha_{D=k|f})$
- $n_{D|f} = \sum_i \alpha_{D=i|f}$ is effective sample size

Foundations 2b – Special Cases to compute $\bar{v}_{c|e}(.\,|\,.)$

Naïve Bayes

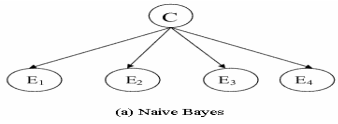
- Class Nodes

$$\bar{v}_{c|e}(C \mid \{\}) = \sum_{c'} \frac{R(c'; e)}{\bar{\theta}_{c'|\{\}}}$$

- Evidence Nodes

$$\bar{v}_{c|e}(E \mid c) = R(c; e)\left[\frac{1}{\bar{\theta}_{e|c}} - 1\right]$$

$$\bar{v}_{c|e}(E \mid c') = \left[P_{\bar{\Theta}}(c \mid e) P_{\bar{\Theta}}(c' \mid e)\right]^2 \left[\frac{1}{\bar{\theta}_{e|c'}} - 1\right]$$



(a) Naive Bayes

where
c = value mentioned in query,
c' = any other possible value of C
e = value of evidence **E** appearing in query

$$R(c; \mathbf{e}) = \left[P_{\bar{\Theta}}(c \mid \mathbf{e}) - P_{\bar{\Theta}}(c \mid \mathbf{e})^2\right]^2$$

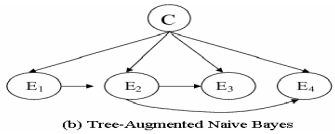
Tree Augmented Naïve Bayes

- Class Nodes
  Evidence Node w/0 evidence parents
  - Same as Naïve Bayes
- Evidence Nodes w/1 evidence parent

$$\bar{v}_{+c|e}(E \mid +f, +c) = R(c; e)\left[\frac{1}{\bar{\theta}_{e|+f,+c}} - 1\right]$$

$$\bar{v}_{+c|e}(E \mid +f, -c) = \left[P_{\bar{\Theta}}(+c \mid e) P_{\bar{\Theta}}(-c \mid e)\right]^2 \times \left[\frac{1}{\bar{\theta}_{e|+f,-c}} - 1\right]$$

$$\bar{v}_{+c|e}(E \mid -f, +c) = \bar{v}_{+c|e}(E \mid -f, -c) = 0$$



(b) Tree-Augmented Naive Bayes

where
value of F matches query $+f$ or not $(-f)$; value of C matches query $+c$ or not $(-c)$

Experiments

How to compute the Best MUV

- 5-fold cross-validation
- 12 UCI data sets … descretized, …
- MUV(*k*NB, ⟨boot/disj, bal/skew⟩, js/mle)

Base Classifiers | How samples are generated | How classifiers are combined

- Also MUV(*k*TAN, ⟨boot/disj, bal/skew⟩, js/mle)

Answer: Use ⟨boot, bal⟩, js

- Bootstrap is better than Disjoint with $p<1.41E-10$
- Balanced is better than Skewed with $p < 3.59E-09$
- James-Stein estimator is better than MLE
  $p<2.3E-5$ [bootstrap+balanced] and
  $p<0.0099$ [disjoint+balanced case]

Comparision with Boosting

- Consider *AdaBoost* : base = Naïve Bayes
- Then… give THESE base-classifiers to MUV…
  MUV(kNB, AdaBoost, js) is better than AdaBoost[NB] with $p < 0.023$

Mixing Methods – Base classifiers

For a response c, classifier *i* produces $\mu_i(c)$ and $\sigma^2_i(c)$

❖ **Take 1**

MaxProb:
return argmax$_c${max$_i \mu_i(c)$}

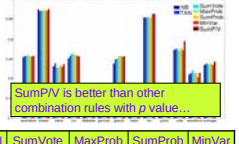
MinVar:
return argmin$_c${min$_i$ $\sigma^2_i(c)$}

❖ **Blur Together**

SumVote:
return argmax$_c${$\Sigma_i\ \delta_i[\mu_i(c) > \mu_i(\text{-}c)]$}

SumProb: (Bagging)
return argmax$_c${$\Sigma_i\ \mu_i(c)$}

SumP/V: (MUV)
return argmax$_c${$\Sigma_i\ \mu_i(c)/\sigma^2_i(c)$}



1NB	1TAN	SumVote	MaxProb	SumProb	MinVar
0.003	0.016	0.015	0.003	0.011	0.001

Mixing Methods – MUV classifiers

- Learn many MUVs and Combine responses from 9 MUVs
  - •1NB, … , 4NB, 1TAN, …, 4TAN, "MUV[Adaboost]" ⟨boot, bal⟩,js

SumP/V[MUV] (a.k.a MUV[MUV]) is better than other combinations with ***p <0.039***

Conclusion

- **MUV** = new ensemble classifier:
  - Instance specific weight, based on variance
  - Use ⟨boot, bal⟩,js
- Very effective
- Very efficient (straight-line code)
- MUV[MUV] is yet better !