Probability calibration methodologies with local expert ensembles

CPT Nick Normandin

Introduction

What should you know about this brief?

- **Please ask questions as I go**

What should you know about this brief?

- **Please ask questions as I go**
- I've assumed some audience proficiency in modern machine learning techniques, but I will alternatively try to provide a heuristic understanding of the concepts presented

What should you know about this brief?

- **Please ask questions as I go**
- I've assumed some audience proficiency in modern machine learning techniques, but I will alternatively try to provide a heuristic understanding of the concepts presented
- Full accompanying R code is available

What should you know about this brief?

- **Please ask questions as I go**
- I've assumed some audience proficiency in modern machine learning techniques, but I will alternatively try to provide a heuristic understanding of the concepts presented
- Full accompanying R code is available
- This work was funded by the Omar N. Bradley Officer Research Fellowship in Mathematics

What is `local expert`?

I created a new kind of ensemble forecasting method that I've called local expert regression. It involves the decomposition of a supervised learning task with a continuous target variable (*regression*) into a series of of many $\{0, 1\}$ mappings corresponding to separate *binary probabilistic classification* tasks that produce estimates on the $[0, 1]$ interval.

Why is this useful ?!

Because you can aggregate the ensemble predictions to form a completely unique *probability distribution function* for each prediction. You can understand **risk** not just in terms of a model, but in terms of each individual forecast.

…see **github.com/nnormandin/localexpeRt**

What problem am I solving?

Most classification methods produce scores for class membership which are interpereted as measures of class affiliation probability. This is the foundation of local expert regression. However, these 'probabilities' are not usually **well-calibrated**. For a model f and score s_i to be well-calibrated for class c_i, the empirical probability of a correct classification $P(c_i|f(c_i|x_i) = s_i)$ must converge to $f(c_i|x_i) = s_i$.

For example, when $s_i = 0.9$, the probability of a correct classification should converge to $P(c_i|s_i = 0.9) = 0.9$. Otherwise, this isn't *really* a 'probability.'

How do I propose to solve it?

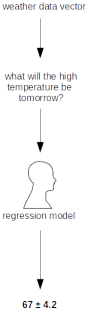
If probabilities aren't properly calibrated, the PDFs interpolated from them won't be reliable. How can we deal with this?

1. Change the loss function
   - $n^{-1} \sum_{i=1}^{n} -y_i \log(p_i) - (1 - y_i) \log(1 - p_i)$
2. Calibrate probabilities
   - isotonic regression, sigmoid transforms?

Local expert

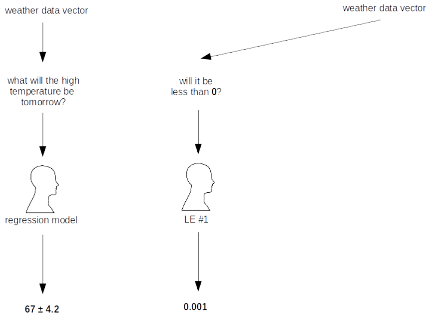
How is local expert different from normal regression?

Let's say you want to predict the high temperature in West Point, NY, tomorrow and you have a vector of covariates $\mathbf{x}_i$ pertaining to weather forecasting.



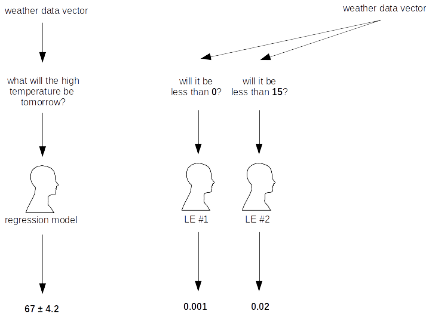
How is local expert different from normal regression?

Let's say you want to predict the high temperature in West Point, NY, tomorrow and you have a vector of covariates $\mathbf{x}_i$ pertaining to weather forecasting.



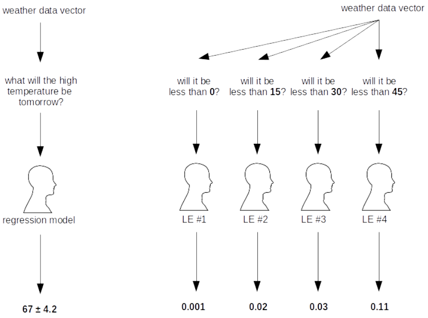
How is local expert different from normal regression?

Let's say you want to predict the high temperature in West Point, NY, tomorrow and you have a vector of covariates $\mathbf{x}_i$ pertaining to weather forecasting.



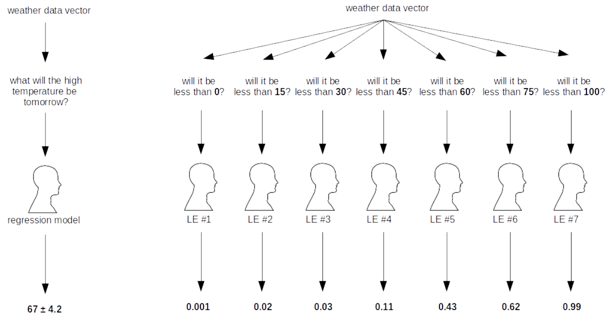
How is local expert different from normal regression?

Let's say you want to predict the high temperature in West Point, NY, tomorrow and you have a vector of covariates $\mathbf{x}_i$ pertaining to weather forecasting.



How is local expert different from normal regression?

Let's say you want to predict the high temperature in West Point, NY, tomorrow and you have a vector of covariates $\mathbf{x}_i$ pertaining to weather forecasting.



Probability calibration

Case study

Results

Conclusion

What have I demosntrated?

Which topics require more research?

1. Compensation for class imbalance
2. Kappa-based optimization methods
3. Incorporation of SMOTE
4. High-level parallelization

Questions