Scaling Up the Accuracy of Naive–Bayes Classifiers: a Decision–Tree Hybrid

Ronny Kohavi

Data Mining and Visualization Group

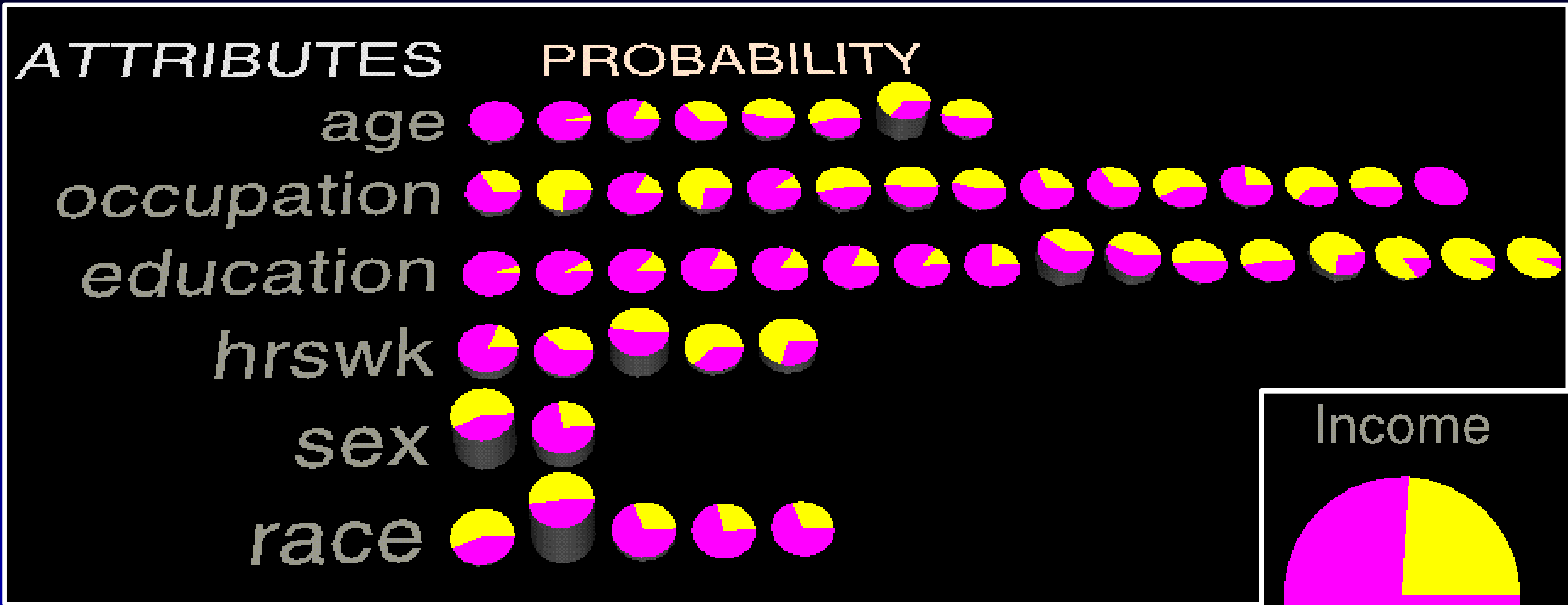
Silicon Graphics, Inc.

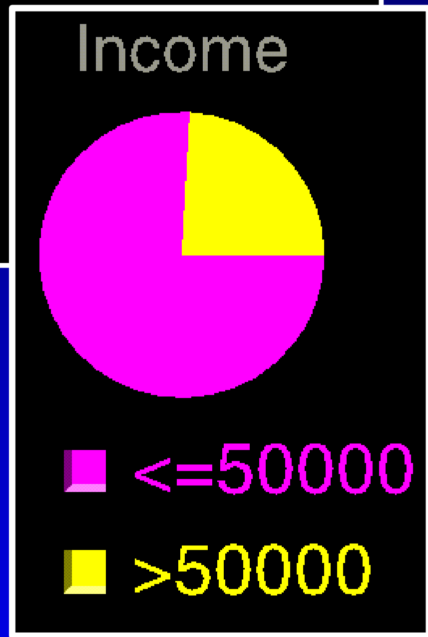
The Naive–Bayes Classifier

- The Naive–Bayes classifier computes the probabilities of each label value given the record, assuming attributes are **conditionally independent** given the label.

- The assumption seems very strong but:
  - Naive–Bayes performs surprisingly well in experiments [Kononko 1993; Langley & Sage 1994; Kohavi & Sommerfield 1995].
  - Correct classification does not require accurate estimates of probabilities [Friedman 1996; Domingos & Pazzani 1996]

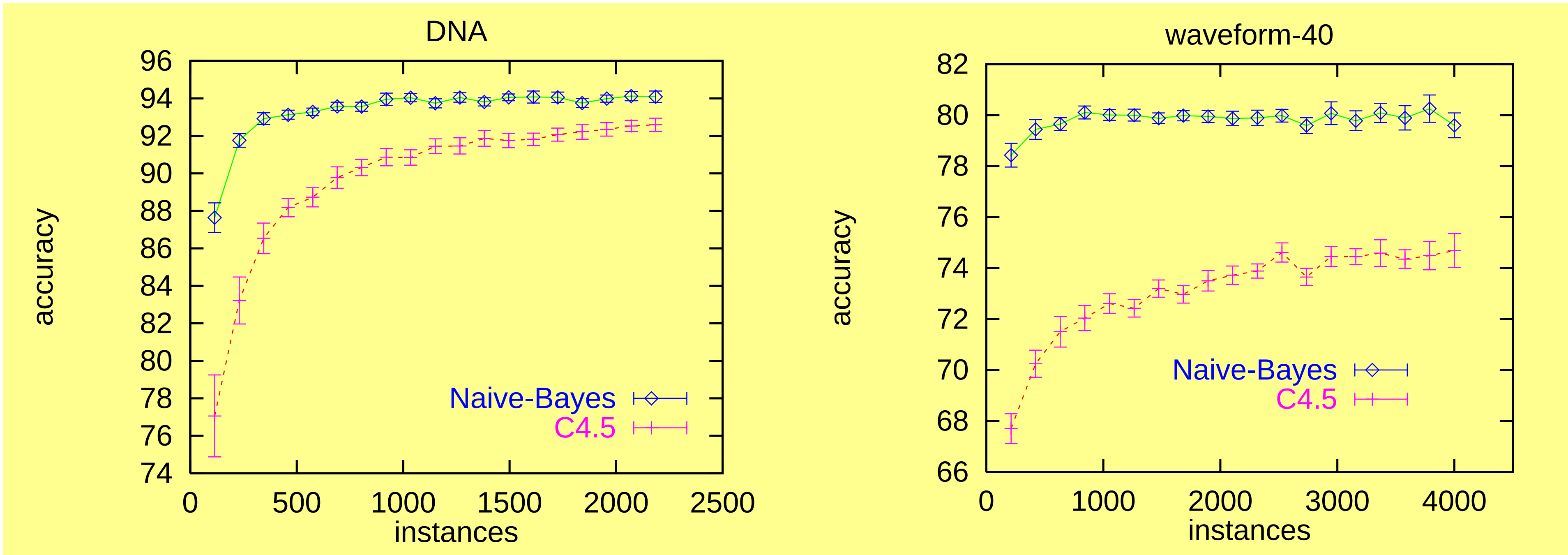
Interpretability





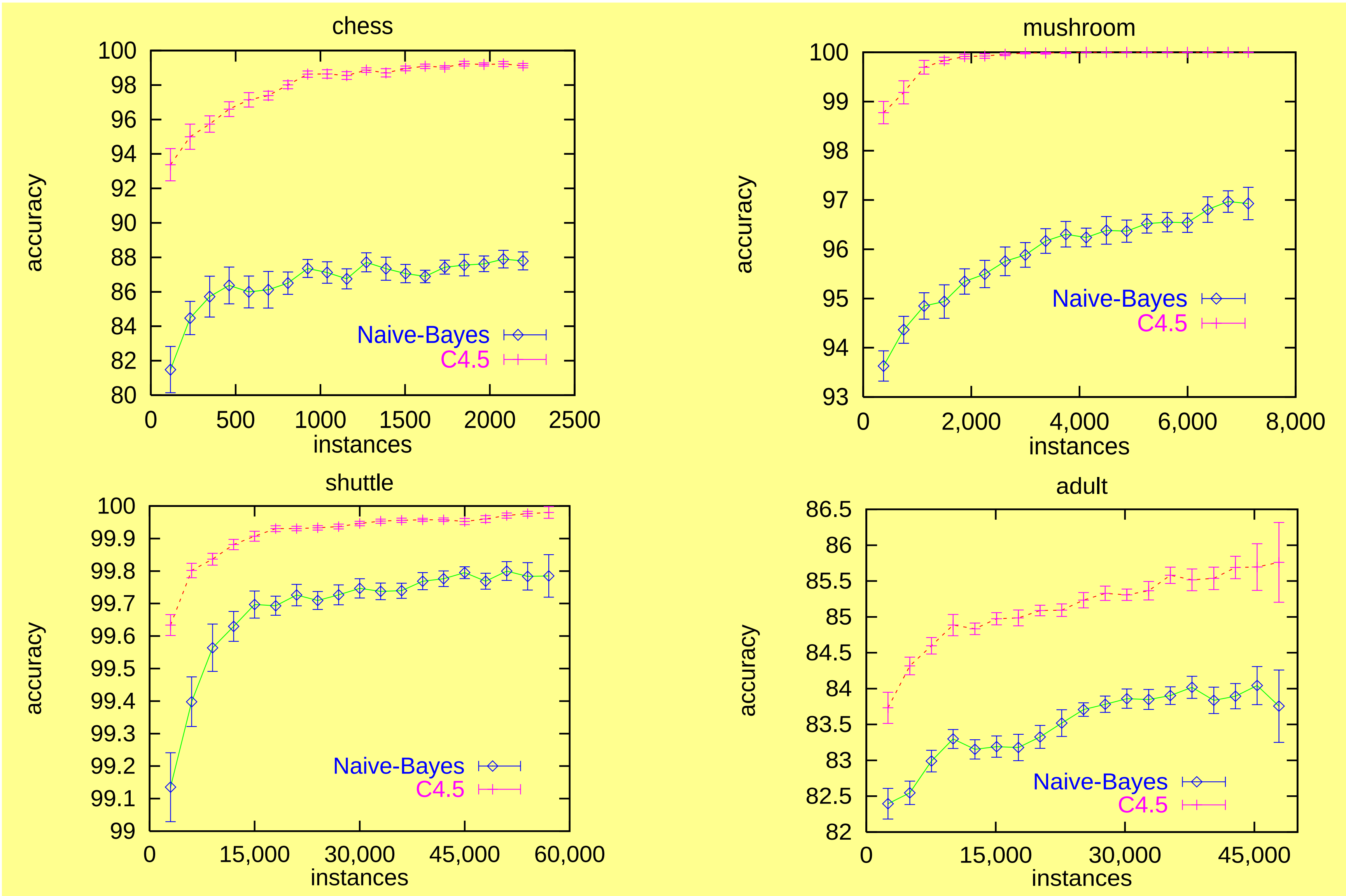
Census Bureau data on working adults in 1994.
Classification: who makes over $50K

Sometimes It Even Scales!

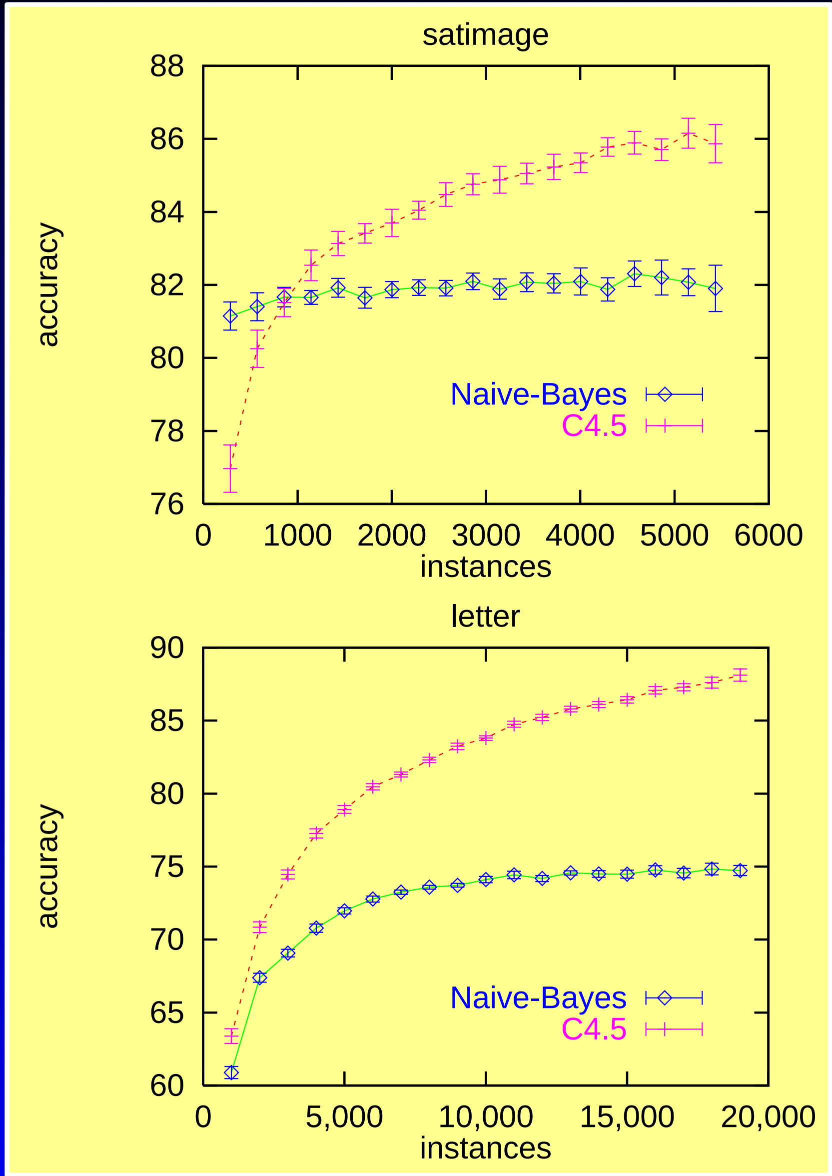


Two semi–large datasets showing Naive–Bayes significantly outperforms C4.5 (decision–trees).

But Often it does Not



And NB Asymptotes Early



A cross–over.

Naive–Bayes starts better but does not improve and asymptotes early.

C4.5 is still improving while Naive–Bayes asymptoted early.

When is Naive–Bayes Better?

- Many irrelevant features. Naive–Bayes is very robust to irrelevant features. The conditional probabilities for irrelevant features equalize (hence do not affect prediction) fast.

- Predictions require taking into account many features. Decision trees suffer from fragmentation in these cases.

- The assumptions hold, i.e., when features are conditionally independent and equally important (e.g., medical domains).

When are Decision–Trees Better?

- Serial tasks: once the value of a key feature is known, dependencies and distributions change. A good example is chess. Another view of this: when segmenting the data into subpopulations gives "easier" subproblems.

- There *are* key features: some features are much more important than others. In the mushroom dataset, the *odor* attribute alone gives you over 98% accuracy. Naive–Bayes never got to this level.

NBTree: a Hybrid

- Use the decision tree to segment the data into subproblems and apply Naive–Bayes to each one.
- Decision nodes will test attributes as with regular decision trees, but the leaves will contain Naive–Bayes classifiers.
- Since NB is good at handling many features with relatively little data, it is used where it is most useful: the leaves.

How to Segment the Data

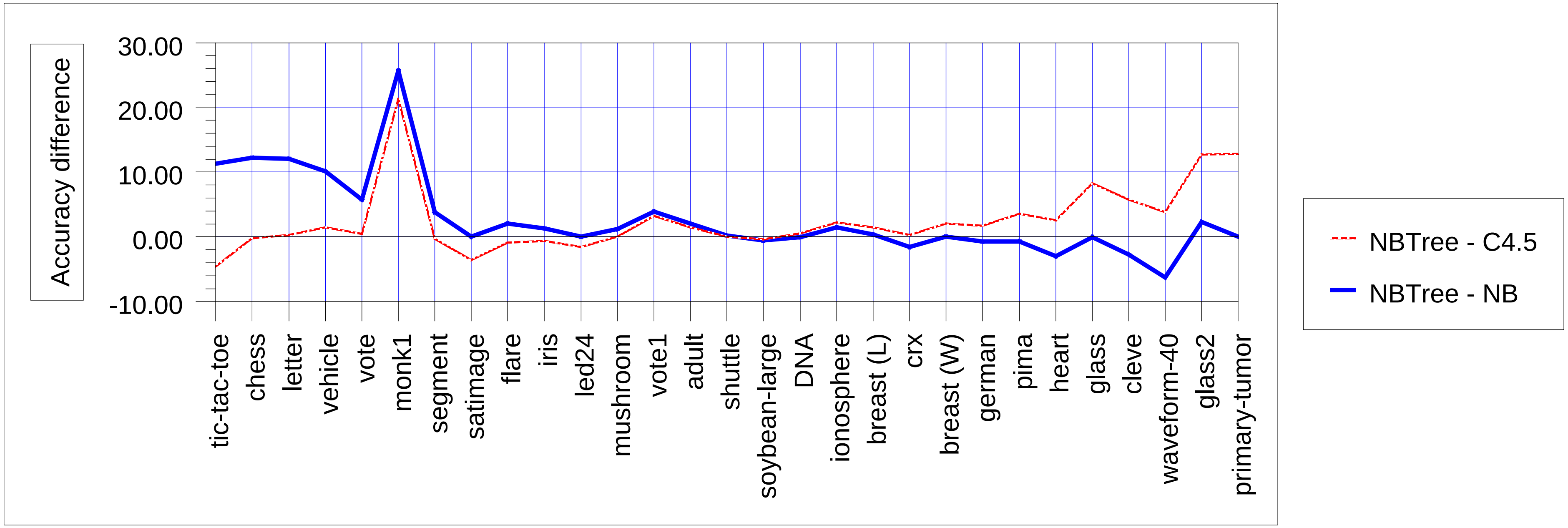
- Observation: Naive–Bayes is an incremental induction algorithm, which means cross–validation can be done fast (linear in the number of instances) by deleting folds, testing them, and inserting them again.

- Instead of finding a direct splitting criteria such as mutual–info/Gini/gain–ratio, we use cross–validation to estimate how much a split would help versus creating an NB–leaf.

  We don’t attempt to fundamentally derive when a split is useful; we try it out.

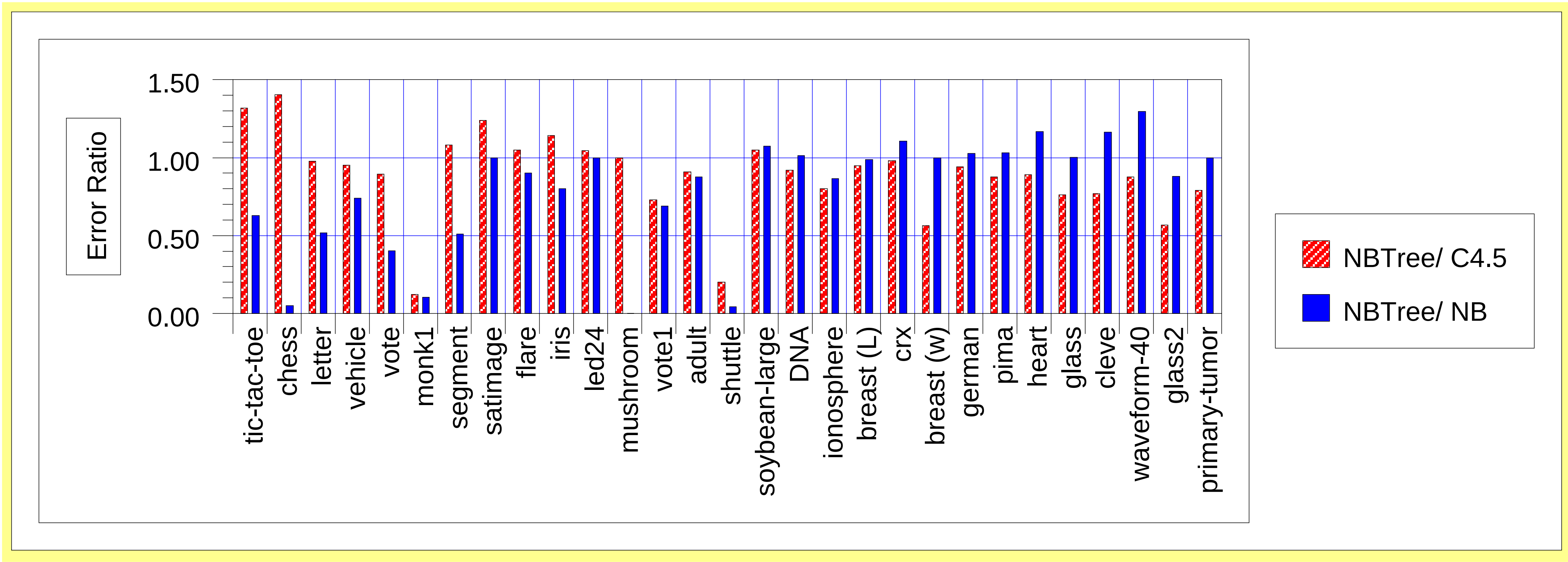
Results: Absolute Differences

Difference in accuracy between NBTree and C4.5, and NBTree and Naive–Bayes. Above the zero lines means NBTree is better.



Results: Relative Differences

Relative difference in accuracy between NBTree and C4.5, and NBTree and Naive–Bayes. Below 1.0 means NBTree is better.



Interpretability

- The resulting structure is relatively easy to interpret.

- While NBTrees have complex leaves, there are fewer nodes overall:

  Letter: 2109 nodes (C4.5) versus 251 (NBTree)
  Adult: 2213 versus 137
  DNA: 31 versus 3
  LED24: 49 versus 1

  Many leaves end up as regular decision tree leaves because they contain a single class.

Summary

- NBTree combines decision tree based segmentation of the data with Naive–Bayes at the leaves.
- Induction time is slower, but the complexity is the same (constants are bigger).
- Scales well: the accuracy is good for large files. On the three largest files (shuttle, adult, letter), NBTree outperformed both C4.5 and Naive–Bayes.