Assessing the predictive performance of machine learners in software defect prediction

Martin Shepperd
Brunel University

martin.shepperd@brunel.ac.uk

Understanding your fitness function!

Martin Shepperd
Brunel University

martin.shepperd@brunel.ac.uk
That ole devil called accuracy (predictive performance)

Martin Shepperd
Brunel University

martin.shepperd@brunel.ac.uk

Acknowledgements

- Tracy Hall (Brunel)
- David Bowes (Uni of Hertfordshire)

Initial Premises

- Lack of deep theory to explain software engineering phenomena
- Machine learners widely deployed to solve software engineering problems
- Focus on one class – fault prediction
- Many hundreds of fault prediction models published [5]

But
- No one approach dominates
- Difficulties in comparing results
Further Premises

- compare models using prediction performance (statistic)
- view as a fitness function
- statistics measure different attributes / may sometimes be useful to apply multi-objective fitness functions
  
  **BUT!**

- need to sort out flawed and misleading statistics

Dichotomous classifiers

- Simplest (and typical) case.
- Recent systematic review located 208 studies that satisfy inclusion criteria [5]
- Ignore costs of FP and FN (treat as equal).

- Data sets are usually highly unbalanced i.e., +ve cases < 10%.
ML in SE Research Method

1. Invent/find new learner
2. Find data
3. REPEAT
4. Experimental procedure E yields numbers
5. IF numbers from new learner(classifier) > previous experiment THEN
5. happy
6. ELSE
7. E’ <- permute(E)
8. UNTIL happy
9. publish

Confusion Matrix

\[
\begin{bmatrix}
TP & FP \\
FN & TN
\end{bmatrix}
\]

- $TP =$ true positives (e.g. correctly predicted as defective components)
- $FN =$ false negatives (e.g. wrongly predicted as defect-free)
- $TP, \ldots$ are instance counts
- $n = TP+FP+TN+FN$
Accuracy

\[ \frac{TP + TN}{n} \]

- Never use this!
- Trivial classifiers can achieve very high 'performance' based on the modal class, typically the negative case.

Precision, Recall and the F-measure

- From IR community
- Widely used
- Biased because they don’t correctly handle negative cases.
**Precision (Specificity)**

\[
\frac{TP}{TP + FP}
\]

- Proportion of predicted positive instances that are correct i.e., True Positive
- Accuracy
- Undefined if TP+FP is zero (no +ves predicted, possible for n-fold CV with low prevalence)

**Recall (Sensitivity)**

\[
\frac{TP}{TP + FN}
\]

- Proportion of Positive instances correctly predicted.
- Important for many applications e.g. clinical diagnosis, defects, etc.
- Undefined if TP+FN is zero (ie only -ves correctly predicted).
**F-measure**

\[
\frac{2 \times R \times P}{R + P}
\]

- Harmonic mean of Recall (R) and Precision (P).
- Two measures and their combination focus only on positive examples /predictions.
- Ignores TN hence how well classifier handles negative cases.

\[
\begin{bmatrix}
TP & FP \\
FN & TN
\end{bmatrix}
\]

**Different F-measures**

- Forman and Scholz (2010)
- Average before or merge?
- Undefined cases for Precision / Recall
- Using highly skewed dataset from UCI obtain F=0.69 or 0.73 depending on method.
- Simulation shows significant bias, especially in the face of low prevalence or poor predictive performance.
Matthews Correlation Coefficient

\[ \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \]

- Uses entire matrix
- easy to interpret (+1 = perfect predictor, 0=random, -1 = perfectly perverse predictor)
- Related to the chi square distribution

Matthews (1975) and Baldi et al. (2000)

Motivating Example (1)

\[
\begin{bmatrix}
10 & 100 \\
10 & 100
\end{bmatrix}
\]

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>220</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.50</td>
</tr>
<tr>
<td>precision</td>
<td>0.09</td>
</tr>
<tr>
<td>recall</td>
<td>0.50</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.15</td>
</tr>
<tr>
<td>MCC</td>
<td></td>
</tr>
</tbody>
</table>
Motivating Example (2)

\[
\begin{bmatrix}
10 & 90 \\
20 & 80
\end{bmatrix}
\]

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>200</td>
</tr>
<tr>
<td>accuracy</td>
<td>0.45</td>
</tr>
<tr>
<td>precision</td>
<td>0.10</td>
</tr>
<tr>
<td>recall</td>
<td>0.33</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.15</td>
</tr>
<tr>
<td>MCC</td>
<td></td>
</tr>
</tbody>
</table>

Matthews Correlation Coefficient

![Histogram of MCC values]
MCC Highlights Perverse Classifiers

- 26/600 (4.3%) of results are negative
- 152 (25%) are < 0.1
- 18 (3%) are > 0.7
Hall of Shame!!

- The lowest MCC value is actually -0.50
- Paper reported:

```
<table>
<thead>
<tr>
<th>Model</th>
<th>Project</th>
<th>Code</th>
<th>UML</th>
<th>Code</th>
<th>UML</th>
<th>Code</th>
<th>UML</th>
</tr>
</thead>
<tbody>
<tr>
<td>ECR</td>
<td>80%</td>
<td>80%</td>
<td>100%</td>
<td>100%</td>
<td>67%</td>
<td>67%</td>
<td></td>
</tr>
<tr>
<td>CRNS</td>
<td>57%</td>
<td>64%</td>
<td>80%</td>
<td>80%</td>
<td>6%</td>
<td>25%</td>
<td></td>
</tr>
<tr>
<td>BNS</td>
<td>33%</td>
<td>67%</td>
<td>50%</td>
<td>75%</td>
<td>0%</td>
<td>50%</td>
<td></td>
</tr>
</tbody>
</table>
```

and concluded:

Despite our encouraging findings, external validity has not been fully proved yet, and further empirical studies are needed, especially with real data from the industry.

Hall of Shame (continued)

- A paper in TSE (65 citations) has MCC = -0.47, -0.31
- Paper reported:

```
<table>
<thead>
<tr>
<th>Observed as</th>
<th>Classified as</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0/1</td>
</tr>
<tr>
<td></td>
<td>0/2</td>
</tr>
<tr>
<td>FF</td>
<td>NFP</td>
</tr>
<tr>
<td>NFP</td>
<td>68</td>
</tr>
<tr>
<td>Total</td>
<td>95</td>
</tr>
</tbody>
</table>
```

and concluded:

The results show that our approach produces statistically significant estimations and that our overall modeling method performs no worse than existing techniques.
Misleading performance statistics

C. Catal, B. Diri, and B. Ozumut. (2007) in their defect prediction study give precision, recall and accuracy (0.682, 0.621, 0.641).

From this Bowes et al. compute an F-measure of 0.6501 [0,1]

But MCC is 0.2845 [-1,+1]
Area Under the Curve

Issues with AUC

- Reduce tradeoffs between TPR and FPR to a single number
- Straightforward where curve A strictly dominates B -> \( \text{AUC}_A > \text{AUC}_B \)
- Otherwise problematic when real world costs unknown
Further Issues with AUC

- Cannot be computed when no +ve case in a fold.
- Two different ways to compute with CV (Forman and Scholz, 2010).
  - WEKA v 3.6.1 uses the AUC$_{merge}$ strategy in its Explorer GUI and Evaluation core class for CV, but AUC$_{avg}$ in the Experimenter interface.

So where do we go from here?

- Determine what effects we (better the target users) are concerned with?
  - Multiple effects?
- Informs fitness function
- Focus on effect sizes (and large effects)
- Focus on effects relative to random
- Better reporting
References


