Week3

Evaluation metrics for binary classifiers

- Represent (Exctract and select features) → Train (fit estimator to data) → Evaluate → Refine cycle (feature and model).
- The metric for train and evaluation don't have to be the same.

Classification - Accuracy with imbalanced classes:

- Accuracy = #correct predictions / #total instances.
- It's useless with imbalanced classes, a dummy classifier (for example one that predicts the majority class) can be similar to a model like SVM.

Dummy classifier

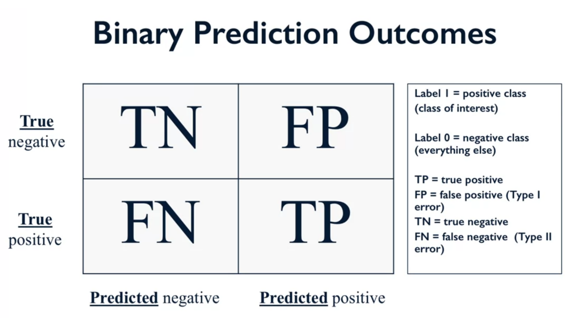
- Provides a null metric (accuracy) baseline.
- To be used for model comparison, not for real data problems outcome.
- Serves as a sanity check on performance.
- There are various strategies for the dummy classifier:

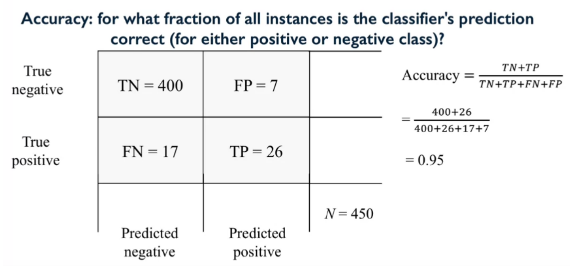
- most_frequent: predicts the most frequent label in the training set.
- stratified: random predictions based on the training set class distribution
- uniform: generates predictions uniformly at random.
- constant always predicts a constant label provided by the user.
    - Useful with F1 score when the positive class is in the minority.
- If my classifier accuracy is close to the null accuracy baseline, it could indicate:
    - Ineffective, erroneous or missing features.
    - Large class imbalance.
    - Poor choice of kernel or hyperparameters.

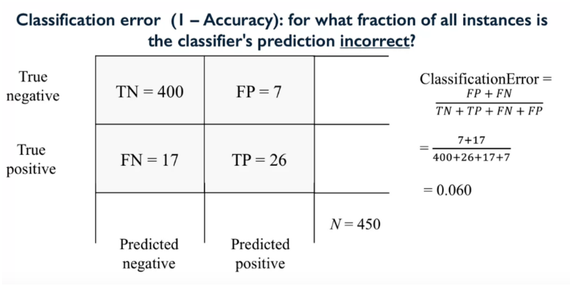
Dummy regressors

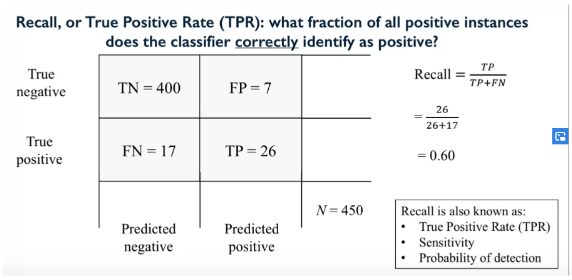
- Same logic, but the strategy turns into:
    - Mean: Predicts the mean of the training targets
    - Median: predicts the median of the training targets.
    - quantile: predicts a user-provided quantile of the training targets.
    - constant: predicts a constant user provided value.

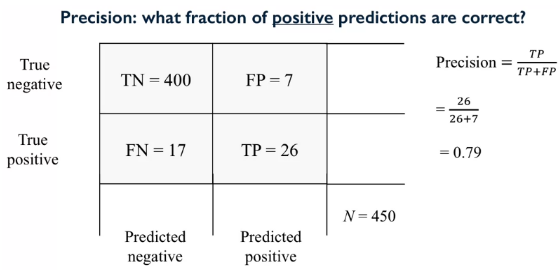
Confusion matrices

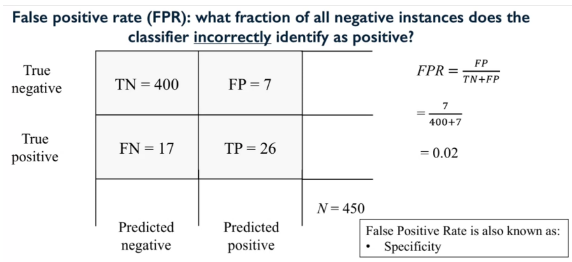












- There is ofter a tradeoff between precision & recall:
  - Recall oriented ML tasks:
    - Search and information extraction in legal discovery
    - Tumour detection
    - Often paired with a human expert to filter out false positives.
  - Precision oriented ML tasks:
    - Document classification.
    - Search engine ranking, query suggestion.
    - Many cusomer-facing tasks

F-score: Combining precision & recall.

- General evaluation metric: **F-score**

$$F_\beta = (1+\beta^2)\frac{Precision.Recall}{(\beta^2 Precision) + Recall} = \frac{(1+\beta^2).TP}{(1+\beta^2).TP + \beta.FN + FP}$$

> 💡 β allows adjustment of the metric to control emphasis on recall vs precision:
> -Precision orientad: β = 0.5 (false positives hurt performance more than false negatives)
> -Recall orientad: β = 2 (false negatives hurt performance more than false positives)

- **F1 score** - Harmonic mean of precision and recall (β = 1):

$$F_1 = 2.\frac{Precision.Recall}{Precision + Recall} = \frac{2.TP}{2.TP + FN + FP}$$

> 💡 In sklearn: **Classification report** provides all this metrics. **Support** is the # instances in the test set that have true label.

Scores for predictions points

Decision functions (decision_function)

- Each classifier score value per test point indicates how confidently the classifier predicts the positive class (large-magnitude positive values) or the negative class (large-magnitude negative values).
- Choosing a fixed decision threshold gives a classification rule.
- By sweeping the decision threshold through the entire range of possible score values, we get a series of classification outcomes that form a curve.

Predicted probability of class membership (predict_proba)

- Typical rule: choose most likely class

  – e.g class 1 if threshold > 0.50.

- Adjusting threshold affects predictions of classifier.
- Higher threshold results in a more conservative classifier

– e.g. only predict Class 1 if estimated probability of class 1 is above 70%

– This increases precision. Doesn't predict class 1 as often, but when it does, it gets high proportion of class 1 instances correct.

• Not all models provide realistic probability estimates

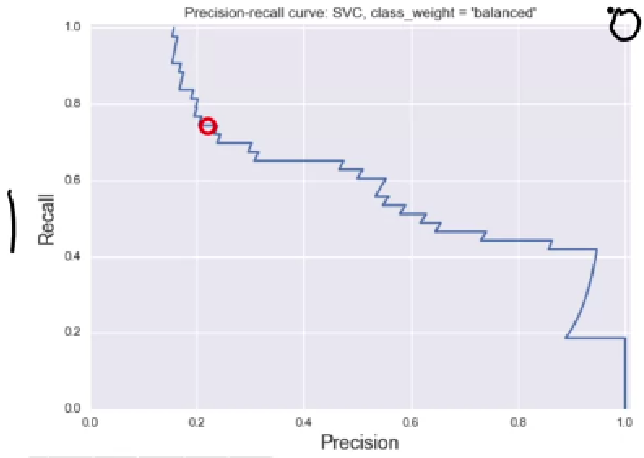
Curves

Precision-Recall

X-axis: Precision
Y-axis: Recall

Top right corner:
- The "ideal" point
- Precision = 1.0
- Recall = 1.0

"Steepness" of P-R curves is important:
- Maximize precision
- while maximizing recall

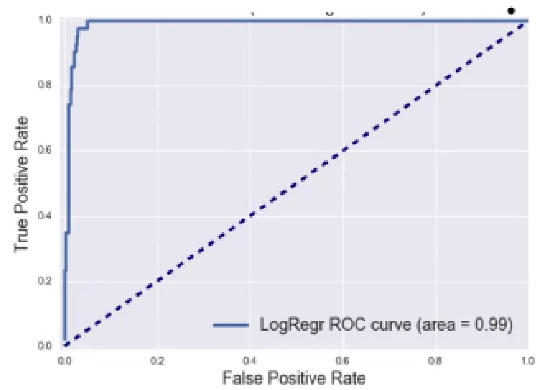


ROC (Receiver Operating characteristic curve)

X-axis: False Positive Rate
Y-axis: True Positive Rate

Top left corner:
- The "ideal" point
- False positive rate of zero
- True positive rate of one

"Steepness" of ROC curves is important:
- Maximize the true positive rate
- while minimizing the false positive rate



- ROC is the graph resulting from plotting **True Positive Rate(sensitivity or recall) vs. False Positive Rate(1 – specificity).**
- We can calculate the Area Under the Curve to summarise a classifier performance.
- Advantages:

- Gives a single number.
- Does not require specifying a decision threshold.

- Disadvantages:
  - As with other single number metric,s AUC loses information (shape of the curve, tradeoffs).

Multi-class evaluation

Extension of binary case

- Multi-class evaluation is an extension of the binary case.

  -A collection of true vs predicted binary outcomes, one per class.

  -Confusion matrices are especially useful

  -Classification report

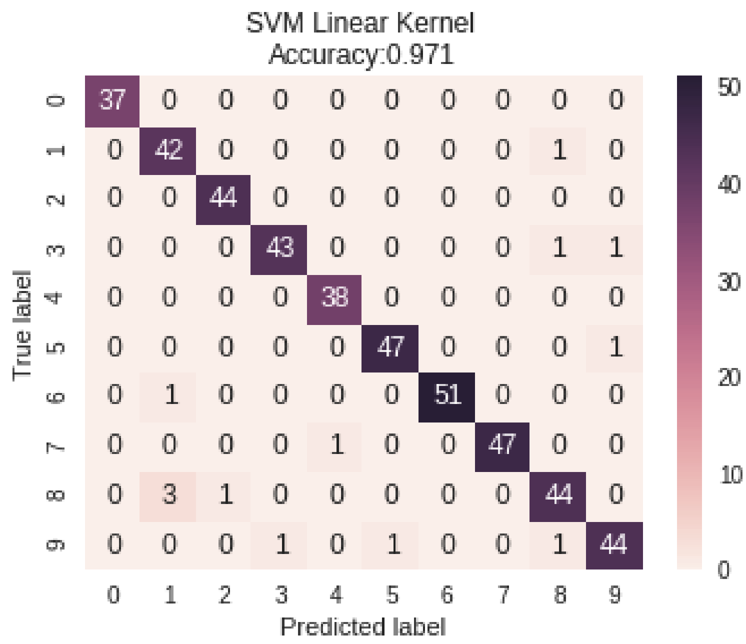
- Overall evaluation metrics are averages across classes

  -But there are different ways to average multi-class results: we will cover these shortly.

  -The support (number of instances) for each class is important to consider, e.g. in case of imbalanced classes

- Multi-label classification: each instance can have multiple labels (not covered here)

Multi-class confusion matrix



Micro vs Macro average

Macro average recall

- Each class has equal weight (doesn't matter it's an imbalanced dataset):

1. Compute the metric (like recall) in each class.
2. Average resulting metrics across classes.

Micro average recall

- Each instance has equal weight.
- Largest classes have more influence.

1. Aggregate outcomes across all classes.
2. Compute metric with aggregate outcomes.

- If the classes have about the same number of instances, macro- and micro-average will be about the same.
- If some classes are much larger (more instances) than others, and you want to:

  -Weight your metric toward the largest ones, use micro-averaging.
  -Weight your metric toward the smallest ones, use macro-averaging.

- If the micro-average is much lower than the macro-average then examine the larger classes for poor metric performance.
- If the macro-average is much lower than the micro-average then examine the smaller classes for poor metric performance.

Regression evaluation

- Typically R2 (computes how well future instances will be predicted) is enough.
  - R2 = 1 is a perfecto model.
  - R2=0 is a model that outputs a constant value regardless of input.
  - R2 can be negative for bad models.
- Alternative:
  - Mean square error (squared differences of target and predicted).
  - Mean absolute error (abs differences of target and predicted).
  - Median absolute error (robust to outliers).
- There are also dummy regressors to establish a baseline. In the case of a simple regression it can be the mean, median, quantile, custom constant , etc.

Model selection

Optimizing classifiers for different evaluation metrics

- Train/Test on same data
  - Single metric
  - Typically overfits.
  - Can serve as sanity check: low accuracy may indicate a model/implementation problem.
- Single train/test split
  - Single metric.

    - Speed and simple.
    - Lack of variance information
- K-fold cross-val
    - K train-test split
    - Average metric over all splits.
    - Can be combined with grid search.
    - It can lead to subtle overfitting/optimistic generalisation
- **So, ideally, use 3 sets.**
    - Training (model building)
    - Validation (model selection)
    - Test (model evaluation).
- So, in practice:
    - Create an initial training/test split.
    - Do cross-validation on the training data for model/parameter selection.
    - Save the held-out test set for final model evaluation

Conclusions

- Accuracy is often not the right evaluation metric for many real-world machine learning tasks:
    - False positives and false negatives may need to be treated very differently.
    - Make sure you understand the needs of your application and choose an evaluation metric that matches your application, user, or business goals.
- Examples of additional evaluation methods include:
    - Learning curve: How much does accuracy (or other metric) change as a function of the amount of training data?
    - Sensitivity analysis: How much does accuracy (or other metric) change as a function of key learning parameter values?