Supervised Regression

$\mathcal{H} = \{ f(\underline{x}) = \underline{\theta}^T \underline{x} \mid \underline{\theta} \in \mathbb{R}^{p+1} \}$

L = L2-Loss

$\mathcal{R}_{emp}(\underline{\theta}) = \sum_{i=1}^{n} (y^{(i)} - \underline{\theta}^T \underline{x}^{(i)})^2$

$\hat{\underline{\theta}} = \arg\min_{\underline{\theta} \in \Theta} \sum_{i=1}^{n} (y^{(i)} - \underline{\theta}^T \underline{x}^{(i)})^2$

$= \arg\min_{\underline{\theta} \in \Theta} \|\underline{y} - X\underline{\theta}\|_2^2 = (X^T X)^{-1} X^T \underline{y}$

Supervised Classification

Scoring Classifier: scoring/discriminant functions $f_1, \dots, f_g : \mathcal{X} \to \mathbb{R}$.

classifier $h(\underline{x}) := \arg\max_{k \in \{1,\dots,g\}} f_k(\underline{x})$

Prob. Classifier: probability functions $\pi_1, \dots, \pi_g : \mathcal{X} \to [0,1]$ with $\sum_{k=1}^{g} \pi_k = 1$.

classifier $h(\underline{x}) := \arg\max_{k \in \{1,\dots,g\}} \pi_k(\underline{x})$

Linear Classifier: $f_1, \dots, f_g$ are called linear specifiers if

$\exists\, g$ rank-preserving, monotone: $g(f_k(\underline{x})) = \underline{w}_k^T \underline{x} + b_k$

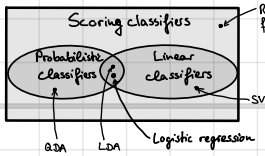
Approaches to construct classifiers

Generative Approach: "Which y tends to have $\underline{x}$ like that?"

$\pi_k(\underline{x}) \propto P(\underline{x} \mid y = k) \cdot \pi_k$ $\quad \pi_k$ being a prior

Discriminant Approach "What is the best prediction for y given $\underline{x}$?".

$\hat{f} = \arg\min_{f \in \mathcal{H}} \mathcal{R}_{emp}(f)$



Examples

Logistic Regression (Discriminant):

$\mathcal{H} = \{ \pi : \mathcal{X} \to [0,1] \mid \pi(\underline{x}) = \frac{\exp(\underline{\theta}^T \underline{x})}{1 + \exp(\underline{\theta}^T \underline{x})} \}$

Discriminant Analysis (Generative)

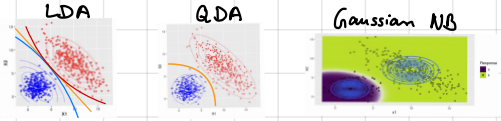
- LDA: assumes $\underline{x} \mid y = k \sim N(\underline{\mu}_k, \Sigma)$ (Linear)
- QDA: assumes $\underline{x} \mid y = k \sim N(\underline{\mu}_k, \Sigma_k)$

Naive Bayes (Generative) assume $p(\underline{x} \mid y = k) = \prod_{j=1}^{p} p(x_j \mid y = k)$

- Gaussian NB: assume $x_j \mid y = k \sim N(\mu_{jk}, \sigma_{jk}^2)$.

  $\Rightarrow \underline{x} \mid y = k \sim N(\underline{\mu}_k, D_k)$ $\quad D_k$ diagonal matrix.
- multinomial NB: assume $p(x_j \mid y = k) \propto \prod_{m=1}^{g} p_{kjm}^{n_{jm}}$

  p_{kjm} = rel. freq. of m in feature j restricted to class k



Overfitting and Underfitting

Underfitting occurs when a model can't reflect the true shape of underlying function (given the data)

↳ High Test Error and high Training Error.

Overfitting occurs when a model reflects noise or artifacts from D_{train} which do not generalize.

↳ Small Train Error and high Test Error.

Overfitting is influenced by:

Complexity $\dim(\Theta)$, n_{train}, $\dim(\mathcal{X})$, aleatoric uncertainty

Overfitting can be avoided by regularization:

less complex model, quality of training data, Feature Selection/Engineering, Occam's razor

Train Error vs. Test Error

Goodness-of-fit measures - like R^2, $\mathcal{L}$, AIC, BIC, deviance - is based on training error but are based on distributional assumptions and limited data - therefore hard to use for high-dimensional or more complex data.

Decrease of n_{train} $\Rightarrow$ Increase of Test Error (in general) b.c. model generalises better with more training data and worse with less training data.

Increase of complexity $\Rightarrow$ Decrease of Training Error (in general) b.c. it becomes easier to learn all patterns on small training data sets or with more flexibility.

Decrease of n_{test} $\Rightarrow$ Increase of variance of test error.

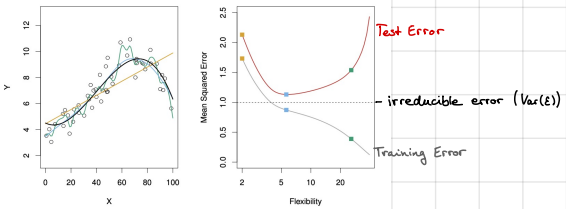
Bias-Variance-Tradeoff

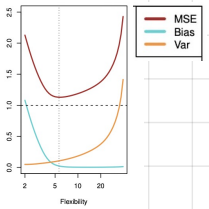
Influence of n_{train}:

Because in practice $n = |D|$ is fixed increase/decrease of n_{train} results in a decrease/increase of n_{test}.

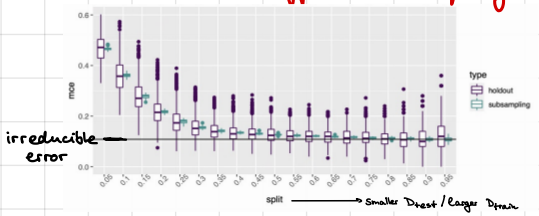
Increasing n_{train} leads to decrease in Test Error but higher variance of Test Error.

Influence of complexity:





Bias-Variance-Trade off in Subsampling



! $\text{Bias}(\text{Subsampling} \mid \text{split-rate}) \doteq \text{Bias}(\text{Hold-out} \mid \text{split-rate})$

$\text{Variance}(\text{Subsampling} \mid \text{split-rate}) < \text{Variance}(\text{Hold-out} \mid \text{split-rate})$

$\Rightarrow$ "optimal" split-rate is higher in Subsampling compared to one Hold-out-split.

Bias-Variance Analysis in Resampling

If there exists a dedicated Test set, then

$\widehat{GE}(\hat{f}, L) = \frac{1}{m} \sum_{(x,y) \in D_{test}} L(y, \hat{f}(\underline{x}))$

$\mathbb{E}[\widehat{GE}(\hat{f}, L)] = GE(\hat{f}, L)$ and $\mathbb{V}[\widehat{GE}(\hat{f}, L)] = \frac{1}{m} \mathbb{V}[L(y, \hat{f}(\underline{x}))]$

We can apply the CLT to approx. the distr. of $\widehat{GE}(\hat{f}, L)$.

$\mathbb{E}[\widehat{GE}(I, J, \rho_L)] \approx GE(I, n_{train}, \rho_L) \leq GE(I, n, \rho)$

ROC Analysis

		True condition: Condition positive	True condition: Condition negative	Prevalence = $\frac{\sum \text{Condition positive}}{\sum \text{Total population}}$	Accuracy (ACC) = $\frac{\sum \text{True positive} + \sum \text{True negative}}{\sum \text{Total population}}$
Predicted condition	Predicted condition positive	True positive, Power	False positive, Type I error	Positive predictive value (PPV), Precision = $\frac{\sum \text{True positive}}{\sum \text{Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\sum \text{False positive}}{\sum \text{Predicted condition positive}}$
	Predicted condition negative	False negative, Type II error	True negative	False omission rate (FOR) = $\frac{\sum \text{False negative}}{\sum \text{Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\sum \text{True negative}}{\sum \text{Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\sum \text{True positive}}{\sum \text{Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\sum \text{False positive}}{\sum \text{Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate = $\frac{\sum \text{False negative}}{\sum \text{Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\sum \text{True negative}}{\sum \text{Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$	F_1 score = $\frac{1}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}} = \frac{2\,TP}{2\,TP + FN + FP}$

! Not a good performance measure when label distr. is unbalanced. Accuracy paradox

→ Usually preferable in situations with unbalanced data.

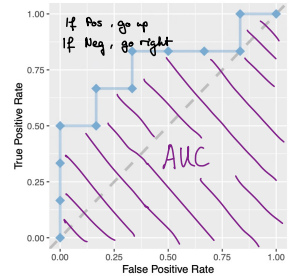
→ Balances TPR and PPV but tends to the smaller value.

Different metrics emphasize different aspects of performance. Choice requires Domain Knowledge.

DRAWING ROC CURVES

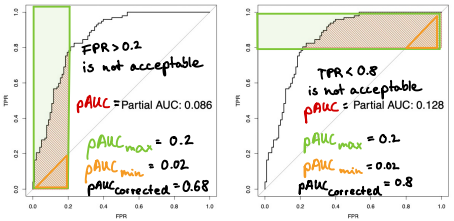
Sorted

#	Truth	Score
1	Pos	0.95
2	Pos	0.86
3	Pos	0.69
4	Neg	0.65
5	Pos	0.59
6	Neg	0.52
7	Pos	0.51
8	Neg	0.39
9	Neg	0.28
10	Neg	0.18
11	Pos	0.15
12	Neg	0.06



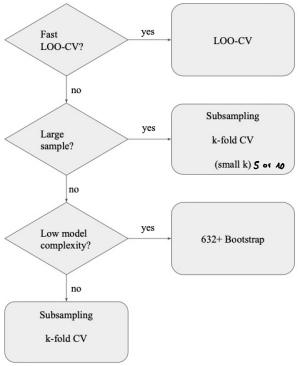
The scores are given by the classifier for each observation. Step-size is determined by $\frac{\#\text{Pos}}{\#\text{Total}}$ and $\frac{\#\text{Neg}}{\#\text{Total}}$

Partial AUC



$pAUC_{corrected} = \frac{1}{2}\left(1 + \frac{pAUC - pAUC_{min}}{pAUC_{max} - pAUC_{min}}\right)$

Guidelines



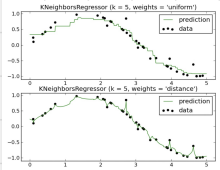
k-NN

Prediction Regression

$$\hat{f}(\underline{x}) = \frac{1}{k} \sum_{i \in \mathcal{I}_k(\underline{x})} y^{(i)}$$

$$\hat{f}(\underline{x}) = \frac{1}{\sum_{i \in \mathcal{I}_k(\underline{x})} w^{(i)}} \cdot \sum_{i \in \mathcal{I}_k(\underline{x})} w^{(i)} y^{(i)}$$

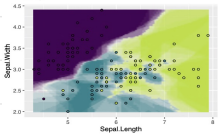
with $w^{(i)} = \frac{1}{d(\underline{x}^{(i)}, \underline{x})}$



Prediction Classification

$$\hat{\pi}_\ell(\underline{x}) = \frac{1}{k} \sum_{i \in \mathcal{I}_k(\underline{x})} \mathbb{1}[y^{(i)} = \ell]$$

$$\hat{h}(\underline{x}) = \underset{\ell \in \{1,\dots,g\}}{\arg\max}\ \hat{\pi}_\ell(\underline{x})$$



Standardization and Weights

- Features are usually standardized or normalized.
  because most distances place higher importance on features with higher ranges.
- Features can be given a higher importance by using a weighted distance measure.

$$d^{\text{weighted}}_{\text{Euclidean}}(\underline{x}, \underline{\tilde{x}}) = \sqrt{\sum_{j=1}^{p} w_j (x_j - \tilde{x}_j)^2}$$

Measures for Regression

Pointwise outer-Loss

Mean squared error (MSE): $\rho_{MSE}(\vec{y}, F) = \frac{1}{m} \cdot \sum_{i=1}^{m} (y^{(i)} - \hat{y}^{(i)})^2$

Mean absolute error (MAE): $\rho_{MAE}(\vec{y}, \vec{F}) = \frac{1}{m} \cdot \sum_{i=1}^{m} |y^{(i)} - \hat{y}^{(i)}|$

Mean abs. perc. error (MAPE): $\rho_{MAPE}(\vec{y}, \vec{F}) = \frac{1}{m} \cdot \sum_{i=1}^{m} \left| \frac{y^{(i)} - \hat{y}^{(i)}}{y^{(i)}} \right|$

Set-based outer-Loss

R^2: $\rho_{R^2}(\vec{y}, \vec{F}) = 1 - \frac{\rho_{MSE}(\vec{y}, LM(\vec{y}))}{\rho_{MSE}(\vec{y}, \bar{y})}$

! Higher $R^2 \not\Rightarrow$ Better fit.
But R^2 is invariant w.r.t. linear scaling of y. MSE is not.

Generalized R^2:

$1 - \frac{\text{Loss}_{\text{ComplexModel}}}{\text{Loss}_{\text{SimpleModel}}}$. E.g. model vs. constant, linear vs. non-linear, tree vs. forest, ...

! Usually $R^2 \in [0,1]$, but this is only true if we evaluate on Training Data. On Test Data $R^2 < 0$ is possible → Overfitting

Measures for Classification

Pointwise outer-Loss using class-labels

Misclassification error rate (MCE):

$$\rho_{MCE} = \frac{1}{m} \cdot \sum_{i=1}^{m} [y^{(i)} \neq \hat{y}^{(i)}] = \frac{\sum \text{False pos.} + \sum \text{False neg.}}{\text{Total population}}$$

Accuracy (ACC):

$$\rho_{ACC} = \frac{1}{m} \cdot \sum_{i=1}^{m} [y^{(i)} = \hat{y}^{(i)}] = \frac{\sum \text{True pos.} + \sum \text{True neg.}}{\text{Total population}}$$

! No information about how good/skewed prob's are.
Errors on all classes weighted equally, which is often inappropriate

Cost matrix:

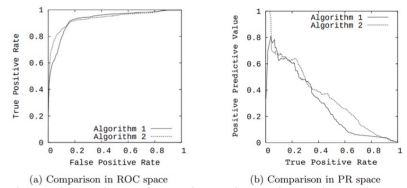
$$\text{Costs} = \frac{1}{n} \cdot \sum_{i=1}^{n} C[y^{(i)}, \hat{y}^{(i)}] = \frac{1}{n} \langle \text{Cost Matrix}, \text{Confusion Matrix} \rangle_F$$

Pointwise outer-Loss using probabilities

Brier Score
Multi-Class Brier Score } See Cheat-Sheet
Log Loss
! Take sum/mean

Precision-Recall Curves

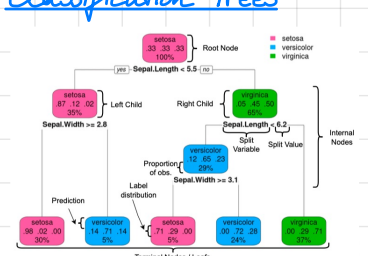
Create a ROC-like plot, but with TPR and PPV instead of TPR and FPR. might be better for highly imbalanced data ($n_- \gg n_+$)

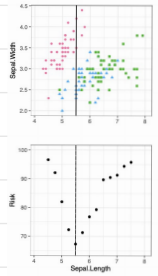


(a) Comparison in ROC space (b) Comparison in PR space

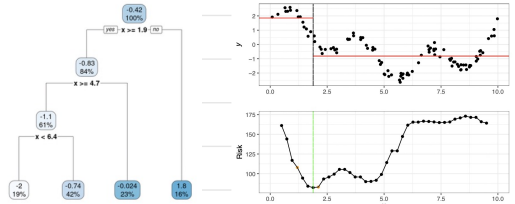
Curve dominates fully in ROC ⇔ Curve dominates fully in PR.

Classification Trees





Regression Trees



Dealing with Categorical Features

If x_j is categorical with m levels, then there are $2^{m-1}-1$ possible partitions.

- binary classification:
  1. Calculate the proportion of 1-outcomes for each category of the feature in $\mathcal{N}$.
  2. Sort the categories according to these proportions.
  3. The feature can then be treated as if it was ordinal, so we only have to investigate at most $m-1$ splits.
- regression with L2-Loss:
  1. Calculate the mean of the outcome in each category
  2. Sort the categories by increasing mean of the outcome
  3. The feature can then be treated as if it was ordinal, so we only have to investigate at most $m-1$ splits.

Proximities

- Imputing missing data:
  1. Replace missing values for a given variable using the median of the non-missing values
  2. Get proximities
  3. Replace missing values in observation $\mathbf{x}^{(i)}$ by a weighted average of non-missing values, with weights proportional to the proximity between observation $\mathbf{x}^{(i)}$ and the observations with the non-missing values

  Steps 2 and 3 are then iterated a few times.
- Locating outliers:
  - An outlier is an observation whose proximities to all other observations are small
  - Measure of outlyingness can be computed for each observation in the training sample
  - If the measure is unusually large, the observation should be carefully inspected
- Identifying mislabeled data:
  - Instances in the training data set are sometimes labeled ambiguously or incorrectly, especially in "manually" created data sets.
  - Proximities can help in finding them: they often show up as outliers in terms of their proximity values.
- Visualizing the forest:
  - The values $1 - \text{prox}(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})$ can be thought of as distances in a high-dimensional space
  - They can be projected onto a low-dimensional space using metric multidimensional scaling (MDS)
  - Metric multidimensional scaling uses eigenvectors of a modified version of the proximity matrix to get scaling coordinates

Random Forests

We use CART's as base learners $b^{[m]}(\underline{x})$ for our bagging.

For each tree: make the tree large and don't use early stopping or pruning
⇒ Increases $\mathbb{V}_{\underline{x},y}[b^{[m]}(\underline{x})]$

At each node: randomly select $\text{mtry} \leq p$ candidate features to consider for splitting
⇒ Decreases $\text{Corr}_{\underline{x},y}[b^{[m]}(\underline{x}), b^{[m]}(\underline{x})]$

- For regression: $\text{mtry} = \lfloor \frac{p}{3} \rfloor$
- For classification: $\text{mtry} = \lfloor \sqrt{p} \rfloor$

Out-of-Bag Prediction/Error

$$OOB^{[m]} := \{ i \in \{1,\dots,n\} \mid (\underline{x}^{(i)}, y^{(i)}) \text{ is out-of-bag for } b^{[m]} \}$$

$$S^{(i)}_{OOB} := \sum_{m=1}^{M} \mathbb{1}(i \in OOB^{[m]}) \quad \text{– amount of trees for which the } i\text{-th observation is OOB.}$$

$$\hat{y}^{(i)}_{OOB} = \begin{cases} \frac{1}{S^{(i)}_{OOB}} \sum_{m=1}^{M} \mathbb{1}(i \in OOB^{[m]}) \cdot \hat{y}^{(i)[m]} & \text{, in regression (scalar)} \\ \left[\frac{1}{S^{(i)}_{OOB}} \sum_{m=1}^{M} \mathbb{1}(i \in OOB^{[m]}) \cdot \mathbb{1}(\hat{h}^{(i)[m]} = k) \right]_{k \in \{1,\dots,g\}} & \text{, in classification (probab. vector)} \end{cases}$$

$$\widehat{err}_{OOB} := \frac{1}{n} \sum_{i=1}^{n} L(y^{(i)}, \hat{y}^{(i)}_{OOB}) \qquad \mathbb{P}(i \in OOB^{[m]}) = (1 - \tfrac{1}{n})^n \to \tfrac{1}{e} \approx 0.37$$

Neural Networks

$\mathcal{H} = \{ f : \mathbb{R}^p \to \mathbb{R} \mid f(\underline{x}) = \sigma\left(\sum_{j=1}^{p} w_j x_j + b\right), w_j \in \mathbb{R}^p, b \in \mathbb{R} \}$ (identity or sigmoid function (Linear regr. or Logistic regr.))

As Loss we use L2 (Lin. regr.) or Bernoulli Loss (Log. regr.)