Classification

WEEK 3

Outline

Classification overview

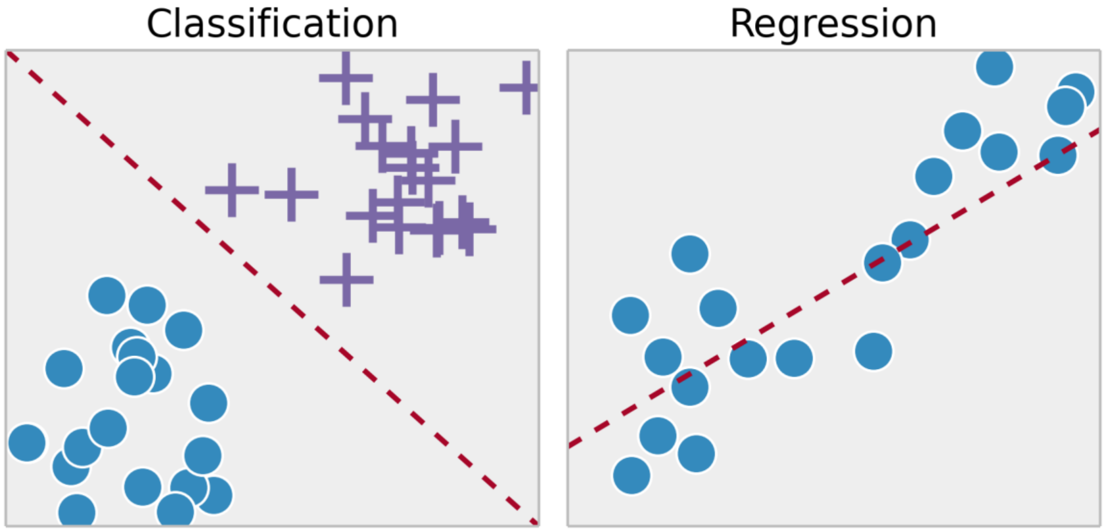
Evaluation on classification models/results

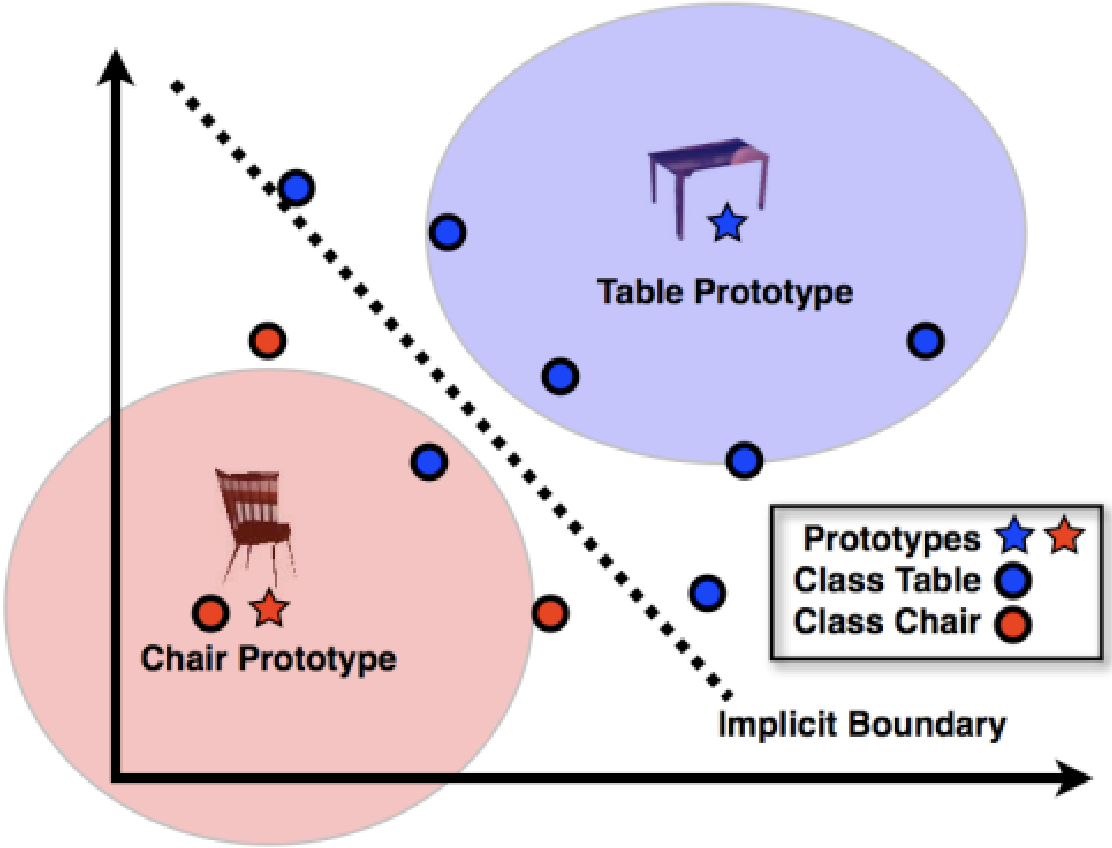
Basic classification models

Ensemble models

Statistical Classification

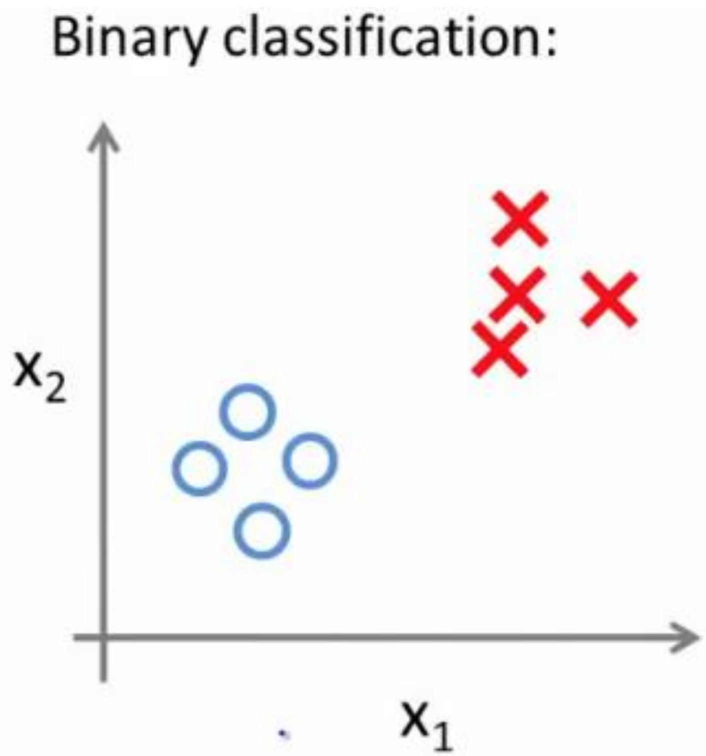
Identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known.

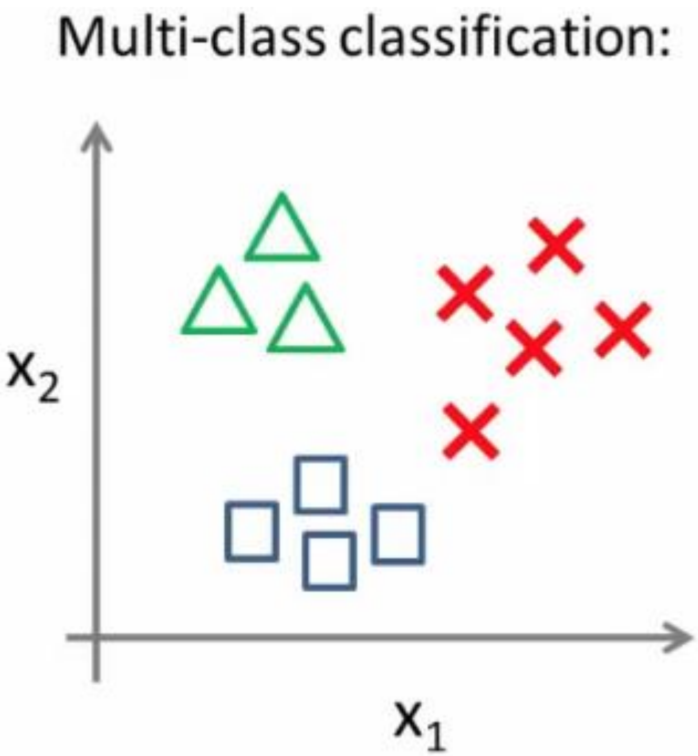




Classification Types (binary vs. multi-class)

Multi-class could be treated as multiple binary classes, focus on binary first



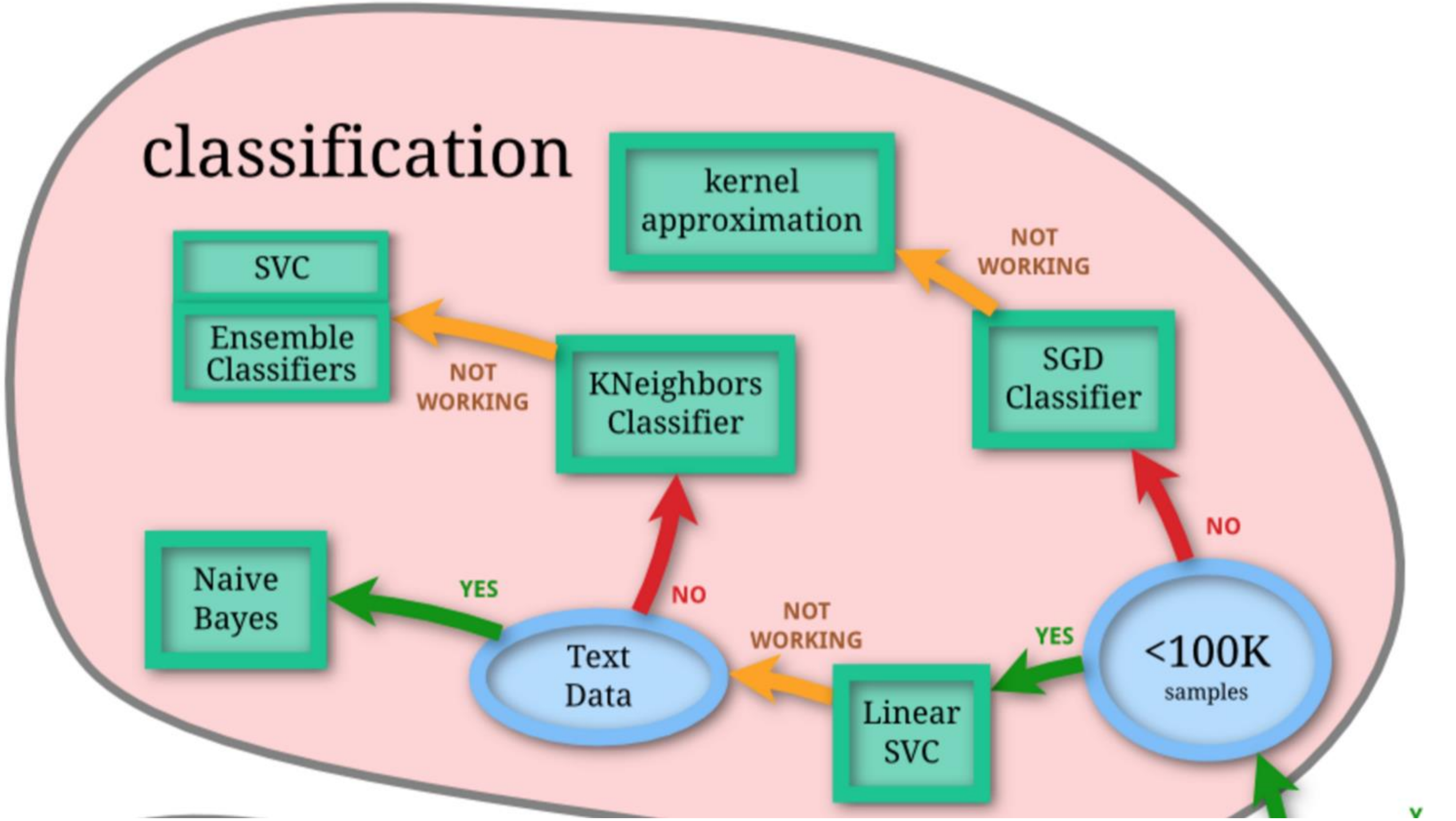


Classification Algorithms

Algorithm have their own pros and cons.

Main one:

Complexity vs. **Generality**



Model Evaluation - General

Know which one works better, for the specific question.

Hold-out

Randomly divide dataset into:

1) training 2) validation 3) test

Cross-Validation

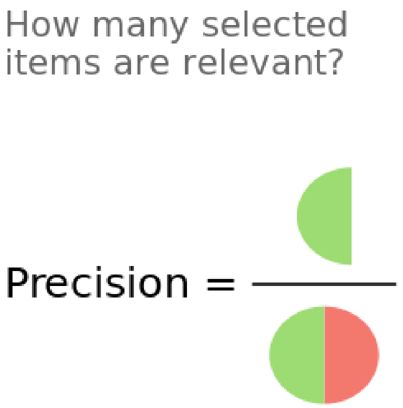
If data is limited

Model Evaluation – Scikit Learn Functions

metrics.accuracy_score(y_true, y_pred[, ...])	Accuracy classification score.
metrics.auc(x, y[, reorder])	Compute Area Under the Curve (AUC) using the trapezoidal rule
metrics.average_precision_score(y_true, y_score)	Compute average precision (AP) from prediction scores
metrics.brier_score_loss(y_true, y_prob[, ...])	Compute the Brier score.
metrics.classification_report(y_true, y_pred)	Build a text report showing the main classification metrics
metrics.confusion_matrix(y_true, y_pred[, ...])	Compute confusion matrix to evaluate the accuracy of a classification
metrics.f1_score(y_true, y_pred[, labels, ...])	Compute the F1 score, also known as balanced F-score or F-measure
metrics.fbeta_score(y_true, y_pred, beta[, ...])	Compute the F-beta score
metrics.hamming_loss(y_true, y_pred[, classes])	Compute the average Hamming loss.
metrics.hinge_loss(y_true, pred_decision[, ...])	Average hinge loss (non-regularized)
metrics.jaccard_similarity_score(y_true, y_pred)	Jaccard similarity coefficient score
metrics.log_loss(y_true, y_pred[, eps, ...])	Log loss, aka logistic loss or cross-entropy loss.
metrics.matthews_corrcoef(y_true, y_pred)	Compute the Matthews correlation coefficient (MCC) for binary classes
metrics.precision_recall_curve(y_true, ...)	Compute precision-recall pairs for different probability thresholds
metrics.precision_recall_fscore_support(...)	Compute precision, recall, F-measure and support for each class
metrics.precision_score(y_true, y_pred[, ...])	Compute the precision
metrics.recall_score(y_true, y_pred[, ...])	Compute the recall
metrics.roc_auc_score(y_true, y_score[, ...])	Compute Area Under the Curve (AUC) from prediction scores
metrics.roc_curve(y_true, y_score[, ...])	Compute Receiver operating characteristic (ROC)
metrics.zero_one_loss(y_true, y_pred[, ...])	Zero-one classification loss.
metrics.brier_score_loss(y_true, y_prob[, ...])	Compute the Brier score.

Model Evaluation – Confusion Matrix

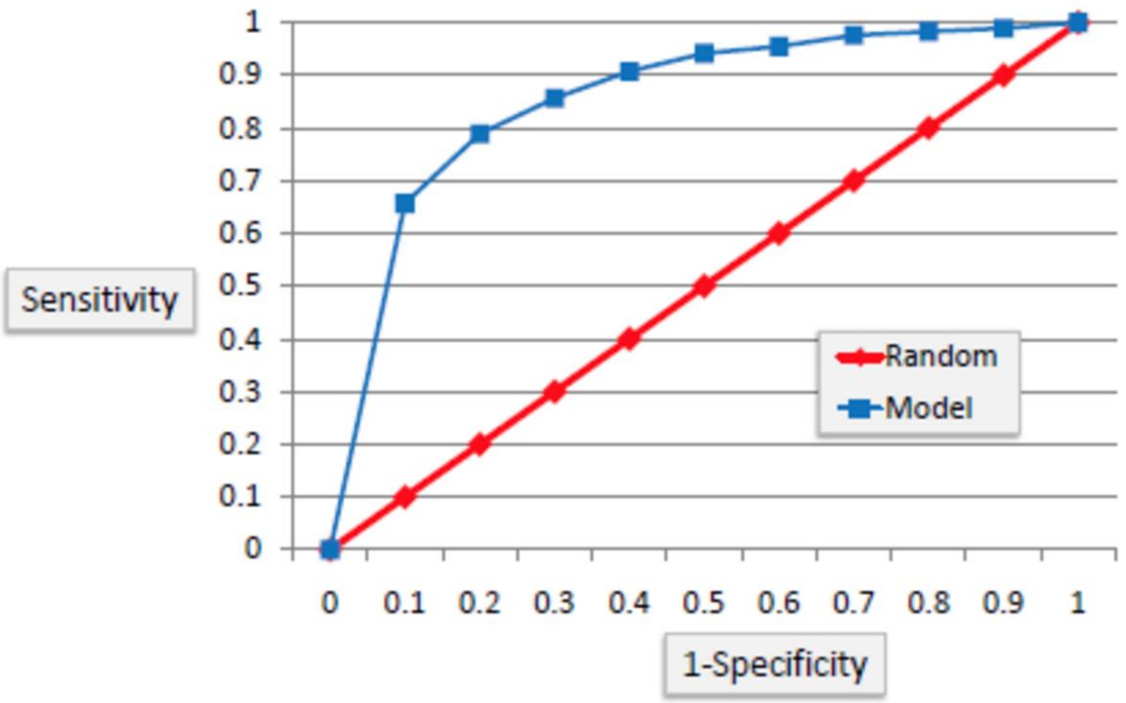
Confusion Matrix		Target			
		Positive	Negative		
Model	Positive	a	b	*Positive Predictive Value*	a/(a+b)
	Negative	c	d	*Negative Predictive Value*	d/(c+d)
		Sensitivity	*Specificity*	**Accuracy** = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		

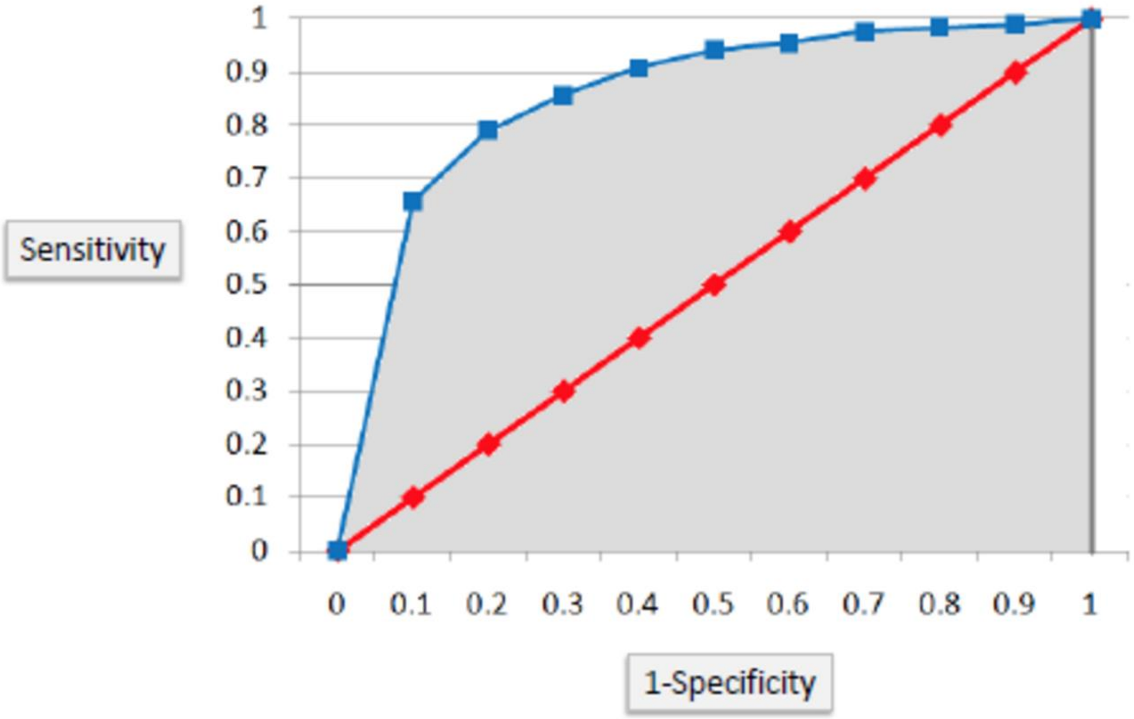


Spam email: 1000 emails, 10 spam; a **spam filter** find out 15 “spams”, 3 out of 15 are real spams

Q: what is the sensitivity (recall), precision, and total accuracy?

Model Evaluation – ROC Curve



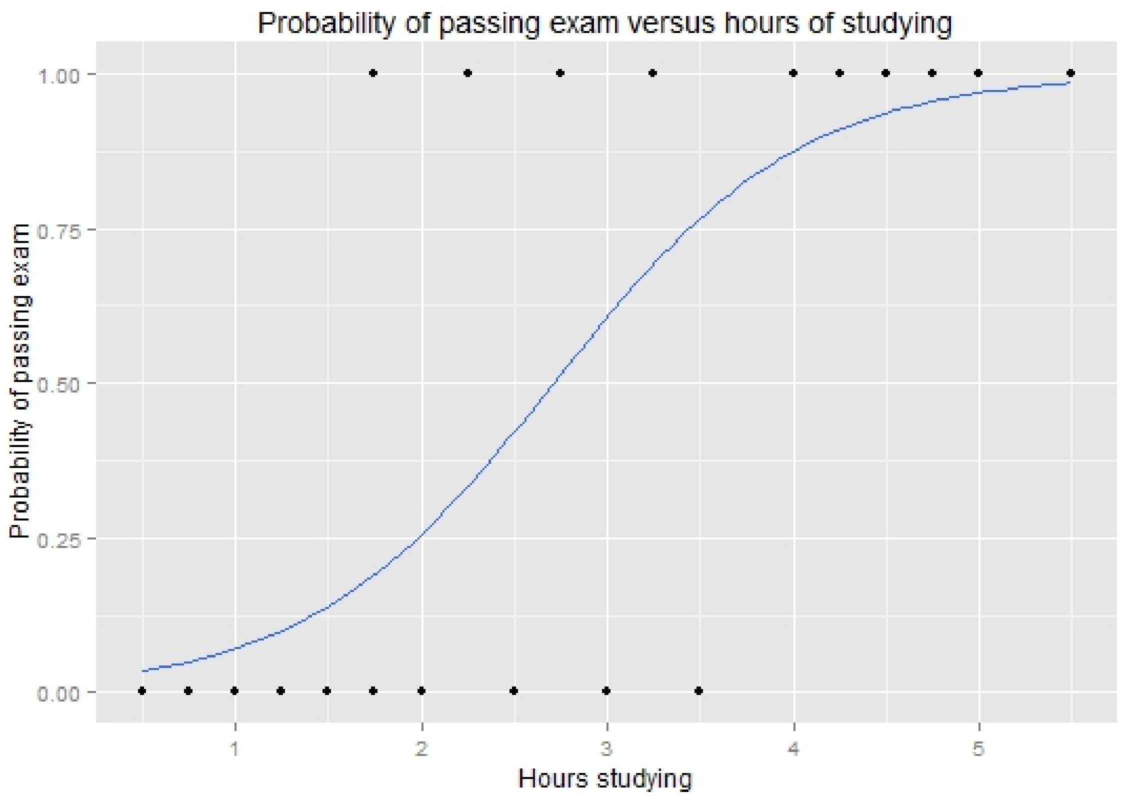


Model: Logistic Regression

Non-linear transformation over linear combination

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Starting with vanilla binary classification (show case cross validation)



Interpreting the coefficient

Log odds ratio

If beta = 1, then:

- x -> x + 1
- log odds ratio change 1
- odds ratio change: 2.718
- P(x+1) / (1-P(x+1)) = 2.718 * P(x) / (1-P(x))

$$P(x) = F(x) = \frac{1}{1+\exp[-(\beta_0+\beta_1 x)]}$$

$$\frac{1}{P(x)} = 1+\exp[-(\beta_0+\beta_1 x)]$$

$$\frac{1-P(x)}{P(x)} = \exp[-(\beta_0+\beta_1 x)]$$

$$\beta_0+\beta_1 x = \log\left[\frac{P(x)}{1-P(x)}\right]$$

$$\frac{P(x)}{1-P(x)} : \text{odds}$$

Model: Logistic Regression

Incorporating L1 and L2 terms, look at the behavior given C.

$$\min_{w,c} \frac{1}{2}w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

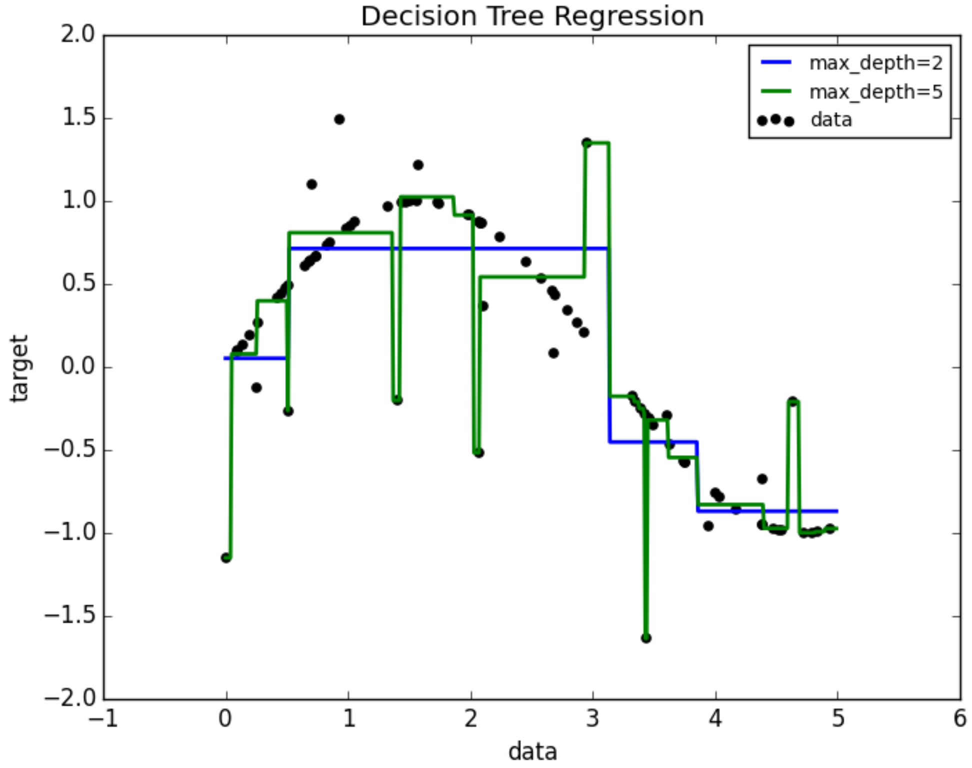
$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \alpha ||w||_1$$

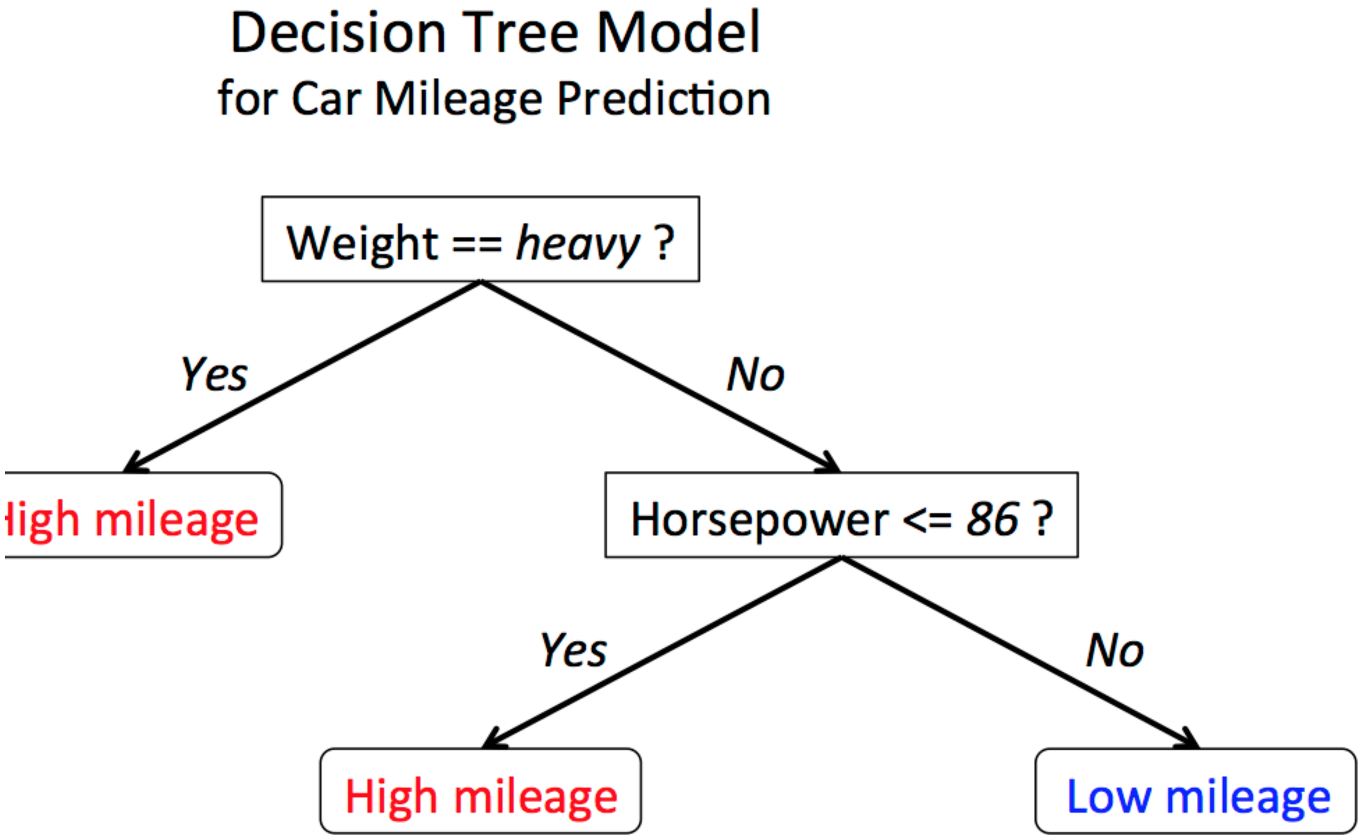
$$\min_{w} ||Xw - y||_2{}^2 + \alpha ||w||_2{}^2$$

$$\min_{w} \frac{1}{2n_{samples}} ||Xw - y||_2^2 + \alpha\rho ||w||_1 + \frac{\alpha(1-\rho)}{2} ||w||_2^2$$

Model: Decision Tree

Naturally non-linear





How decision tree works?

All decision tree models are heuristic (may not be globally optimal)

Three steps:

1. which feature to split?
2. how to split on the selected features?
3. when to stop?

How to measure GOOD or NOT for a split?

Three metrics for split

Classification

- Gini impurity

$$H(X_m) = \sum_k p_{mk}(1 - p_{mk})$$

$$p_{mk} = 1/N_m \sum_{x_i \in R_m} I(y_i = k)$$

- Information gain

$$H(X_m) = -\sum_k p_{mk} \log(p_{mk})$$

Regression

- Mean square error

$$c_m = \frac{1}{N_m} \sum_{i \in N_m} y_i$$

$$H(X_m) = \frac{1}{N_m} \sum_{i \in N_m} (y_i - c_m)^2$$

Example: Gini impurity

Feature value (X1)	Class (Y)
1	A
2	B
3	A
4	B
5	B
6	A

If split at 3.5, there will be two nodes

Before split:

P(A) = 0.50, P(B) = 0.50

After split:

Node 1: A, B, A. P(A) = 0.67, P(B) = 0.33

Node 2: B, B, A. P(A) = 0.33, P(B) = 0.67

Gini impurity (gain)

Before: (all nodes) P(A) * (1-P(A)) + P(B) * (1-P(B)) = 0.5*0.5+0.5*0.5=0.5

After: (% node 1) P(A) * (1-P(A)) + P(B) * (1-P(B)) + (% node 2) P(A) * (1-P(A)) + P(B) * (1-P(B)) =

0.5 * (0.67 * 0.33 + 0.33 * 0.67) + 0.5 * (0.33 * 0.67 + 0.67 * 0.33) = 0.4422

Example: Information gain

Feature value (X1)	Class (Y)
1	A
2	B
3	A
4	B
5	B
6	A

If split at 3.5, there will be two nodes

Before split:

P(A) = 0.50, P(B) = 0.50

After split:

Node 1: A, B, A.   P(A) = 0.67, P(B) = 0.33

Node 2: B, B, A.   P(A) = 0.33, P(B) = 0.67

Information gain

Before: -P(A) * log(P(A)) - P(B) * log(P(B)) = - 0.5 * log(0.5) – 0.5 * log(0.5) = 0.693

After: (% node 1) -P(A) * log(P(A)) - P(B) * log(P(B)) + (% node 2) -P(A) * log(P(A)) - P(B) * log(P(B)) =

0.5 * (-0.67 * log(0.67) - 0.33 * log(0.33) -0.67 * log(0.67) - 0.33 * log(0.33)) = 0.634

Example: where to split? (use Gini)

X1	Y
1	A
2	B
3	A
4	B
5	B
6	A

Split can happen on: 1.5, 2.5, 3.5, 4.5, 5.5.

Before split: P(A) = 0.50, P(B) = 0.50

	Node 1 & 2		Gini 1 & 2	Total Gain
1.5	PA=1.0,PB =0.0, N=1	PA=0.4,PB =0.6, N=5	1/6 * 0 + 5/6 * 0.48 = 0.40	0.5 – 0.4 = 0.1
2.5	PA=0.5,PB =0.5, N=2	PA=0.5,PB =0.5, N=4	2/6 * 0.5 + 4/6 * 0.5 = 0.5	0.5 – 0.5 = 0
3.5	PA=2/3,PB =1/3, N=3	PA=1/3,PB =2/3, N=3	3/6 * 0.44 + 3/6 * 0.44 = 0.44	0.5 - 0.44 = 0.06
4.5	PA=0.5,PB =0.5, N=4	PA=0.5,PB =0.5, N=2	4/6 * 0.5 + 2/6 * 0.5 = 0.5	0.5 – 0.5 = 0
5.5	PA=0.4,PB =0.6, N=5	PA=1.0,PB =0.0, N=1	5/6 * 0.48 + 1/6 * 0 = 0.40	0.5 – 0.4 = 0.1

Example: where to split? (use Gini)

X1	Y
2	B
3	A
4	B
5	B
6	A

Split can happen on: 2.5, 3.5, 4.5, 5.5. Before split: P(A) = 0.40, P(B) = 0.60 => Gini = 0.48

	Node 1 & 2		Gini 1 & 2	Total Gain
2.5	PA=0.0,PB =1.0, N=1	PA=0.5,PB =0.5, N=4	1/5 * 0.0 + 4/5 * 0.5 = 0.4	0.48 – 0.4= 0.08
3.5	PA=0.5,PB =0.5, N=2	PA=1/3,PB =2/3, N=3	2/5 * 0.5 + 3/5 * 0.44 = 0.464	0.48 - 0.464 = 0.016
4.5	PA=1/3,PB =2/3, N=3	PA=0.5,PB =0.5, N=2	3/5 * 0.44 + 2/5 * 0.5 = 0.464	0.48 - 0.464 = 0.016
5.5	PA=1/4,PB =3/4, N=4	PA=1.0,PB =0.0, N=1	4/5 * 0.375 + 1/5 * 0 = 0.3	0.48 – 0.3 = 0.18

Example: where to split? (use Gini)

X1	Y
[redacted]	[redacted]
2	B
3	A
4	B
5	B
6	A

Split can happen on: 2.5, 3.5, 4.5.

Before split: P(A) = 0.25, P(B) = 0.75 => Gini = 0.375

	Node 1 & 2		Gini 1 & 2	Total Gain
2.5	PA=0.0,PB =1.0, N=1	PA=1/3,PB =2/3, N=3	1/4 * 0.0 + 3/4 * 0.44 = 0.33	0.375 – 0.33= 0.045
3.5	PA=0.5,PB =0.5, N=2	PA=0.0,PB =1.0, N=2	2/4 * 0.5 + 2/4 * 0.0= 0.25	0.375 – 0.25= 0.125
4.5	PA=1/3,PB =2/3, N=3	PA=0.0,PB =1.0, N=1	3/4 * 0.44 + 1/4 * 0.0 = 0.33	0.375 – 0.33= 0.045

Example: decision tree result

Feature value (X1)	Class (Y)
1	A
2	B
3	A
4	B
5	B
6	A