Classification

- Classification: Motivation and Applications
- Train-Validation Split and Cross-Validation
- Evaluation Metrics and Class Imbalance
- Overfitting
- kNN Classifier
- Naive Bayes Classifier
- Decision Tree
- Entroy, Conidtional Entropy, Information Gain

Imdad ullah Khan

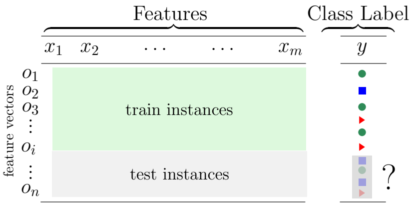
Classification: Definition

Classification is a supervised task

Input: A collection of objects

feature vectors with class labels

Output: A model for the class attribute as a function of other attributes



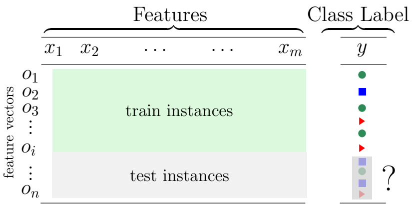
- Training Set: Instances whose class labels are used for learning
- Test Set: Instances with same attributes as training set but missing/hidden class labels
- **Goal:** Model should accurately assign class labels to unlabeled instances

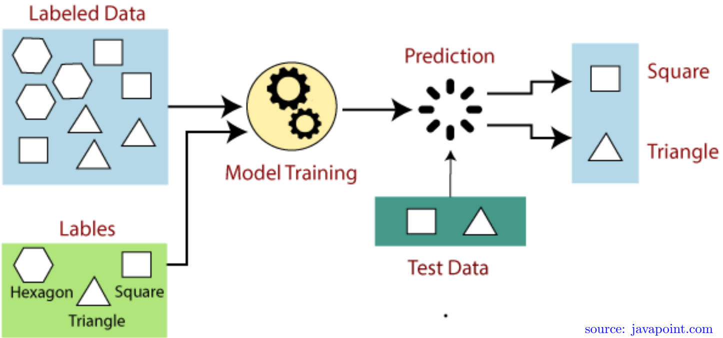
Classification

Input: A collection of objects

feature vectors with class labels

Output: A model for the class attribute as a function of other attributes





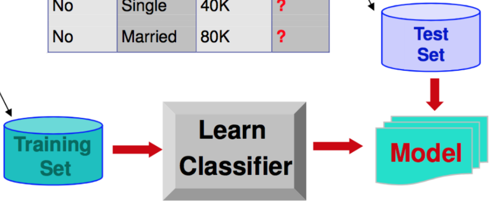
source: javapoint.com

Classification

categorical, categorical, continuous, class

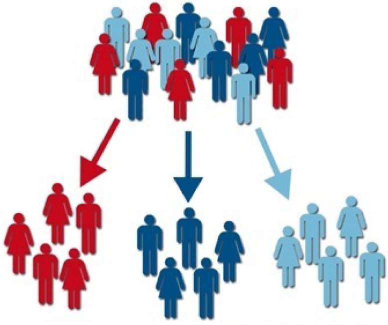
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Refund	Marital Status	Taxable Income	Cheat
No	Single	75K	?
Yes	Married	50K	?
No	Married	150K	?
Yes	Divorced	90K	?
No	Single	40K	?
No	Married	80K	?



Targeted Advertisement

- Enhance marketing by identifying customers who are likely to buy a product
- Use customer purchase history, demographics etc. for similar (old) products
- buy/no buy as class labels

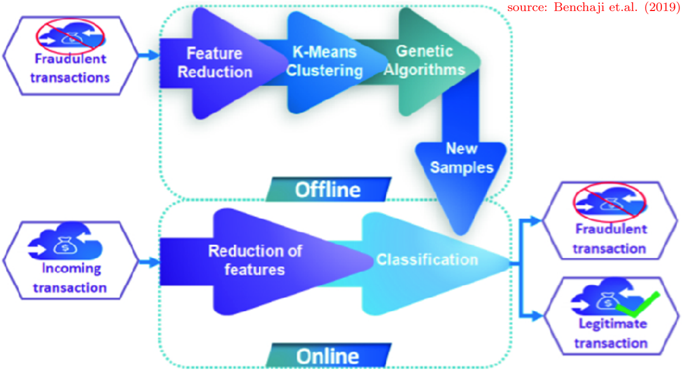


Classification: Applications

Credit Card Fraudulent Transaction Detection

- User transactions history and card holders characteristics
- fair/fraud as class labels

source: Benchaji et.al. (2019)



Classification: Applications

Predict Customer Attrition/Churn

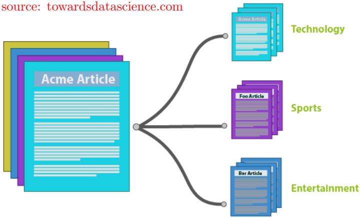
- User customers transactions and feedback history
- churn/no-churn as class labels

Text Classification

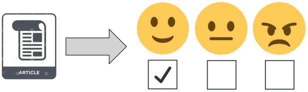
- Text is converted into feature vectors before classification

Document Classification

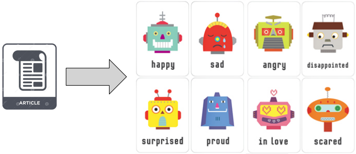
source: towardsdatascience.com



Sentiment Analysis



Emotion Mining



Classification: Applications

Sky Survey Cataloging

- Classify astronomical objects as stars or galaxies
- Use telescoping survey images (from Palomar Observatory)
- 3000 images with 23040×23040 pixels per image
- Extract features values 40 features per object
- star/galaxy as class labels

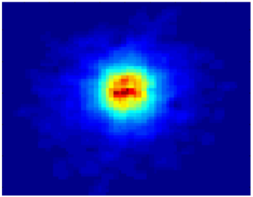
Classification: Applications

Courtesy: http://aps.umn.edu

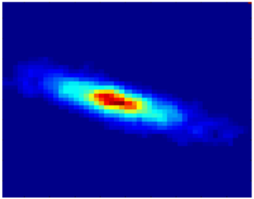
Class:
- **Stages of Formation**

Attributes:
- **Image features,**
- **Characteristics of light waves received, etc.**

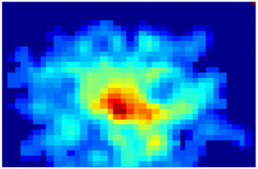
Early



Intermediate



Late



Data Size:
- **72 million stars, 20 million galaxies**
- **Object Catalog: 9 GB**

Classification Evaluation Metrics

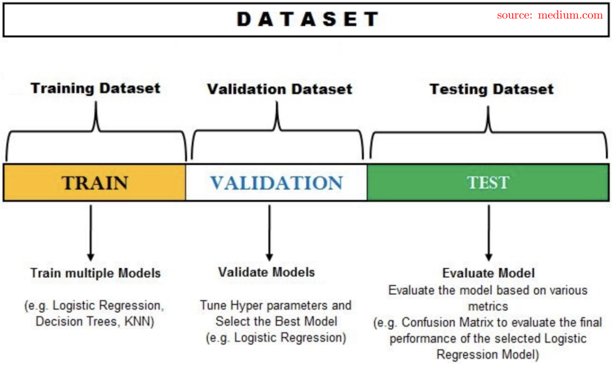
Train-Validation Split and Cross-Validation

Classification

- The model (classifier) is learned by finding patterns in training set
- Performance on training set does not (necessarily) indicate generalization power of the model
- A **validation set** (a subset of training set) is used to learn parameters and tune architecture of classifier and estimate error
- For generalization of the model, validation set must be representative of the input instances
- Since test set is never used during training, it provides an unbiased estimate of generalization error

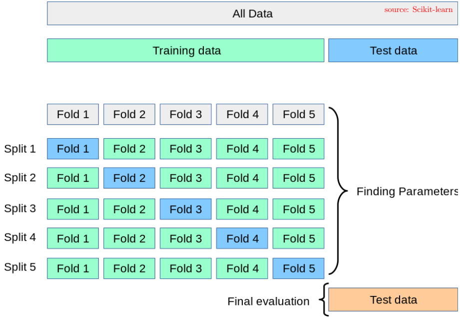
Classification: Training-Validation split

- Generally obtained by randomly splitting the dataset
- e.g. $70-30$, $80-20$ random Train-Validation split
- Use average performance of multiple random (splits)



Classification: Cross-Validation

- The dataset is randomly split into k folds
- In each of k the ith fold is used for validation and the rest for training
- Every instance is used once for validation and $k-1$ times for training
- 5-fold, 10-fold cross-validation



Classification: Evaluation Metrics

Binary Classifiers (for classifying into two classes) are evaluated by tabulating the classification results in a Confusion Matrix

		Actual Classes	
		Positive	Negative
Predicted Classes	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Some summary statistics of the confusion matrix are

$$\text{ACCURACY} = \frac{\text{TP} + \text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} \qquad \text{ERROR} = \frac{\text{FP} + \text{FN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}}$$

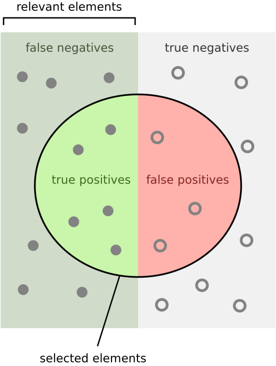
ACCURACY and ERROR are usually reported as percentages

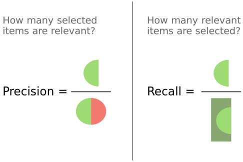
Classification: Evaluation Metrics

		Actual Classes	
		Positive	Negative
Predicted Classes	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

- With big imbalance in classes, ACCURACY and ERROR are misleading
  - In a tumors dataset 99% samples are negative
  - (Blindly) predicting all as negatives gives 99% accuracy
  - But cancer is not detected
- Have to use cost matrix/loss function (essentially weighted accuracy)

Classification: Evaluation Metrics





$$\text{PRECISION} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

▷ sensitivity (measure of exactness)

$$\text{RECALL} = \frac{\text{TP}}{\text{TP}+\text{FN}}$$

▷ specificity (measure of completeness)

- F-measure: Maximizes both
- $F_1 = \dfrac{2}{\dfrac{1}{\text{PRECISION}} + \dfrac{1}{\text{RECALL}}}$
- F_k measure weighs PRECISION and RECALL differently

▷ Weighted harmonic mean

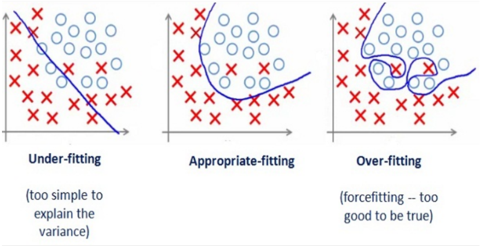
Classification: Overfitting

Overfitting: The phenomenon when model performs very well on training data but does not generalize to testing data

- The model learns the data and not the underlying function

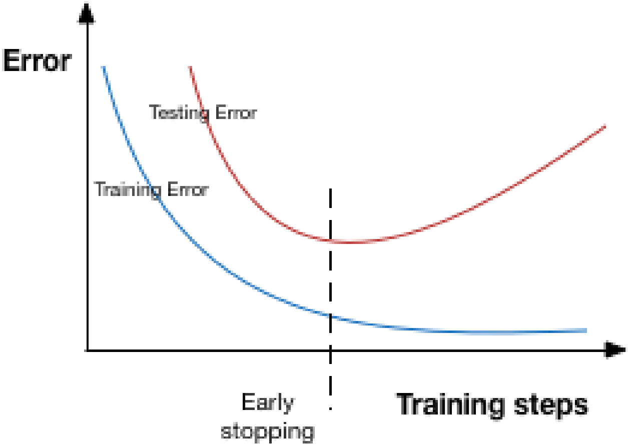
  ▷ Essentially learning by-rote

- Model has too much freedom (many parameters with wider ranges)



- Validation, Cross-validation, early stopping, regularization, model comparison, Bayesian priors help avoiding overfitting

Classification: Overfitting



Classifier/Model

- A classifier utilizes training data to understand how input variables are related to the class variable
- A **model** is built, which can be used to predict labels for unseen data
- Kinds of Classifiers
  - **Lazy Classifiers**
  - **Eager Classifiers**

Kinds of Classifiers

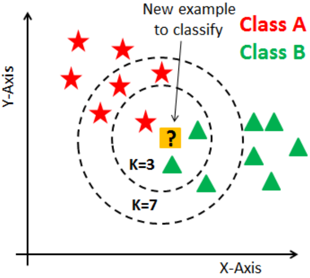
- **Lazy Classifiers**
  - Store the training data and wait for testing data
  - For an unseen test data record (data point), assign class label based on the most related points in the training data
  - Less training time, more prediction time
  - Examples: k-Nearest Neighbor (kNN) Classifier
- **Eager Classifiers**
  - Construct a classification model based on training data
  - For a test data point, use the model to assign class label
  - More training time but less prediction time
  - Examples: Naive Bayes, Decision Tree

Nearest Neighbors Classification and Regression

k-Nearest Neighbor (kNN) Classifier

k-NN is a simple method used for classification ▷ also for regression

The class label of a test instance x is predicted to be the most common class among the k nearest neighbors of x in the train set



- Assign the test instance (?) class A (★) or class B (▲)
- $k = 3$ nearest neighbors (ℓ_2 distance)
  1 ★ and 2 ▲ $\implies$ assigned label = ▲
- $k = 7$ nearest neighbors (ℓ_2 distance)
  4 ★ and 3 ▲ $\implies$ assigned label = ★

k-Nearest Neighbor (kNN)

The class label of a test instance x is predicted to be the most common class among the k nearest neighbors of x in the train set

- Assumes that the proximity measure captures class membership
- Definition of proximity measure (defining 'nearest') is critical
- The parameter k is important and sensitive to local structure of data

k-Nearest Neighbor (kNN) Regression

- In k-NN regression, for a test instance $\mathbf{x}$ the value of target variable y is the 'average' of values of y of k-nearest neighbors of x in train set
- The 'average' can be the weighted mean (weighted by similarity), in this case generally take k so all points are included in neighborhood

$$y(x) \quad = \quad \frac{\sum\limits_{x' \in D} sim(x, x')\, y(x')}{\sum\limits_{x' \in D} sim(x, x')}$$

- $y(x)$ is the value of target variable y in instance x

Naive Bayes Classifier

Naive Bayes Classifier

- Classify $\mathbf{x} = (x_1, \ldots, x_n)$ into one of K classes $C_1, \ldots, C_K$
- Naive Bayes is a conditional probability model
  - For instance $\mathbf{x}$ it computes probabilities $Pr[class = C_j \,|\, \mathbf{x}\,]$ for each class C_j
- Assumes that
  1. All attributes are equally important
  2. Attributes are statistically independent given the class label
     - knowing value of one attribute says nothing about value of another
- Independence assumption is almost never correct (thus the word Naive)

  ▷ but works well in practice
- **Model** is the probabilities calculated from training data for each attribute with respect to class label

Naive Bayes Classifier

Classify $\mathbf{x} = (x_1, \ldots, x_n)$ into one of K classes $C_1, \ldots, C_K$

We want to compute this. The **Posterior probability** of class C_j given the object $\mathbf{x}$

The **Likelihood**: Probability of predictor(s) given a class C_j. Computed from frequencies of predictor(s) in class C_j in train set

Prior: Probability of class C_j, without considering $\mathbf{x}$. Estimated from frequency of labels C_j in train set

$$P(C_j \,|\, \mathbf{x}) = \frac{P(\mathbf{x} \,|\, C_j) \times P(C_j)}{P(\mathbf{x})}$$

Evidence: Probability of observing $\mathbf{x}$, This is indepedent on classes C and $\mathbf{x}$ is given. Effectively constant.

Apply the independence assumption

$$P(\mathbf{x} \,|\, C_j) = P(x_1 \,|\, C_j) \times P(x_2 \,|\, C_j) \times \ldots \times P(x_n \,|\, C_j)$$

Substitute in numerator and ignore the denominator

$$P(C_j \,|\, \mathbf{x}) = P(x_1 \,|\, C_j) \times P(x_2 \,|\, C_j) \times \ldots \times P(x_n \,|\, C_j) \times [\, P(C_j) \,]$$

Naive Bayes: Running Example

Train on records of weather conditions and whether or not game was played.
Given weather condition (test instance) predict whether game will be played

Outlook	Temp.	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

N. Milkic & U. Krcadinac @ Uni. of Belgrade

Naive Bayes: Running Example

$$P(play = \textbf{yes} \,|\, \mathbf{x}) = P(outl = *|\textbf{yes}) \times P(temp = *|\textbf{yes}) \times P(humid = *|\textbf{yes}) \times P(wind = *|\textbf{yes}) \times [P(\textbf{yes})]$$

$$P(play = \textbf{no} \,|\, \mathbf{x}) = P(outl = *|\textbf{no}) \times P(temp = *|\textbf{no}) \times P(humid = *|\textbf{no}) \times P(wind = *|\textbf{no}) \times [P(\textbf{no})]$$

Outlook	Temp.	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	f[illegible]	[illegible]
rainy	cool	normal	t[illegible]	[illegible]
overcast	cool	normal	[illegible]	[illegible]
sunny	mild	high	f[illegible]	[illegible]
sunny	cool	normal	f[illegible]	[illegible]
rainy	mild	normal	f[illegible]	[illegible]
sunny	mild	normal	[illegible]	[illegible]
overcast	mild	high	[illegible]	[illegible]
overcast	hot	normal	f[illegible]	[illegible]
rainy	mild	high	t[illegible]	[illegible]

	Play	
Outlook	**yes**	**no**
sunny	2	3
overcast	4	0
rainy	3	2
TOTAL	9	5

Naive Bayes: Running Example

Outlook	Temp.	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

	Play	
Outlook	yes	no
sunny	2	3
overcast	4	0
rainy	3	2
TOTAL	9	5

For each attribute...

	Play	
Outlook	yes	no
sunny	2	3
overcast	4	0
rainy	3	2
TOTAL	9	5

	Play	
Temp.	yes	no
hot	2	2
mild	4	2
cool	3	1
TOTAL	9	5

	Play	
Humid.	yes	no
high	3	4
normal	6	1
TOTAL	9	5

	Play	
Windy	yes	no
false	6	2
true	3	3
TOTAL	9	5

	Play
yes	9
no	5
TOTAL	14

Naive Bayes: Running Example

	Play			Play			Play			Play			Play
Outlook	**yes**	**no**	**Temp.**	**yes**	**no**	**Humid.**	**yes**	**no**	**Windy**	**yes**	**no**		
sunny	2	3	hot	2	2	high	3	4	false	6	2	yes	9
overcast	4	0	mild	4	2	normal	6	1	true	3	3	no	5
rainy	3	2	cool	3	1								
TOTAL	9	5	TOTAL	9	5	TOTAL	9	5	TOTAL	9	5	TOTAL	14

Covert values to ratios

	Play			Play			Play			Play			Play
Outlook	**yes**	**no**	**Temp.**	**yes**	**no**	**Humid.**	**yes**	**no**	**Windy**	**yes**	**no**		
sunny	0.22	0.60	hot	0.22	0.40	high	0.33	0.80	false	0.67	0.40	yes	0.64
overcast	0.44	0.00	mild	0.44	0.40	normal	0.67	0.20	true	0.33	0.60	no	0.36
rainy	0.33	0.40	cool	0.33	0.20								

$\dfrac{\text{2 occurences of } \textbf{Play = no}\text{, where } \textbf{Outlook = rainy}}{\text{5 occurrences of } \textbf{Play = no}}$

Naive Bayes: Running Example

Given weather condition $\mathbf{x} = (sunny, cool, high, true)$ will game be played?

$$P(play = \mathbf{yes} \,|\, \mathbf{x}) = P(sunny|\mathbf{yes}) \times P(cool|\mathbf{yes}) \times P(high|\mathbf{yes}) \times P(true|\mathbf{yes}) \times [P(\mathbf{yes})]$$

$$= 0.22 \times 0.33 \times 0.33 \times 0.33 \times [0.64] = \mathbf{0.0053}$$

$$P(play = \mathbf{no} \,|\, \mathbf{x}) = P(sunny|\mathbf{no}) \times P(cool|\mathbf{no}) \times P(high|\mathbf{no}) \times P(true|\mathbf{no}) \times [P(\mathbf{no})]$$

$$= 0.60 \times 0.20 \times 0.80 \times 0.80 \times [0.36] = \mathbf{0.0206}$$

	Play			Play			Play			Play			Play
Outlook	**yes**	**no**	**Temp.**	**yes**	**no**	**Humid.**	**yes**	**no**	**Windy**	**yes**	**no**		
sunny	0.22	0.60	hot	0.22	0.40	high	0.33	0.80	false	0.67	0.40	yes	0.64
overcast	0.44	0.00	mild	0.44	0.40	normal	0.67	0.20	true	0.33	0.60	no	0.36
rainy	0.33	0.40	cool	0.33	0.20								

Naive Bayes: Issues

Some issues for Naive Bayes classifier: you are encouraged to read about

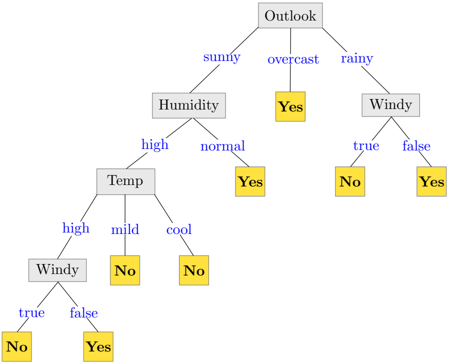
- **Zero frequency problem:** probability $= 0$ for an attribute in a class
  - For example: $P[Outlook = sunny|yes] = 0$
  - One zero would make whole product zero
  - Solution: Laplace smoothing (add-one smoothing)
- Missing value of an attribute for a test instance
  - usually attribute is omitted from probability calculation
- What if values of attributes are continuous?
  - Discretization solves the problem in many cases
  - Can also assume a probability distribution for each continuous attribute and learn distribution parameters from training set

Decision Tree Classifier

Decision Tree

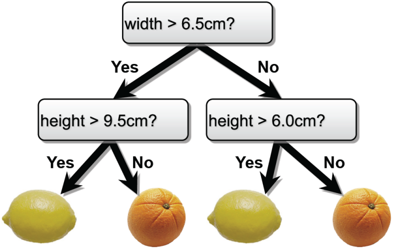
- Fundamentally, an **if-then** rule set for classifying objects
- Builds model in the form of a tree structure

Decision Tree



Decision tree for binary classification of instance with nominal attributes

Zemel, Ustasun, Fidler @ Uni, of Toronto

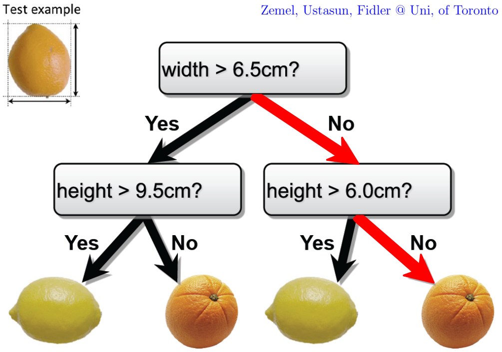


Decision tree for binary classification of instance with numeric attributes

- Each internal node tests an attribute x_i
- Branches correspond to possible (subsets of) values of x_i
- Each leaf node assigns a class label y

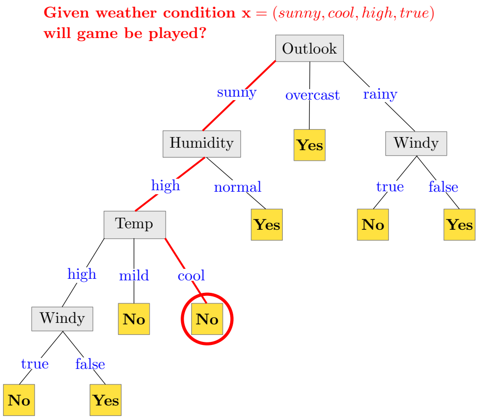
Classification using Decision Trees

- To classify a test instance **x** traverse the tree from root to leaf
- Take branches at internal nodes according to results of their tests
- Predict the class label at the leaf node reached



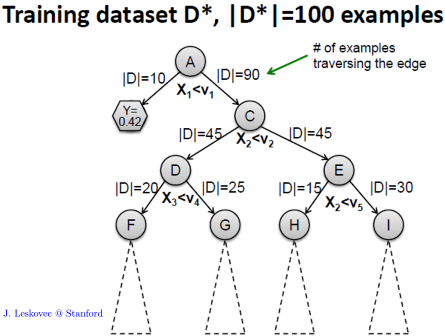
Classification using Decision Trees

- To classify a test instance **x** traverse the tree from root to leaf
- Take branches at internal nodes according to results of their tests
- Predict the class label at the leaf node reached



Building Decision Tree

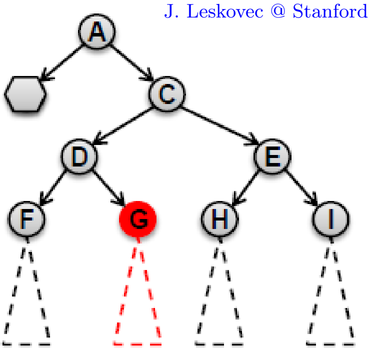
Building the optimal decision tree is NP-HARD problem



- Recursively build the tree top-down, using greedy heuristics
- Start with an empty decision tree
- Split the current dataset by the best attribute until stopping condition

Building Decision Tree

- Suppose at some node G in the tree built so far

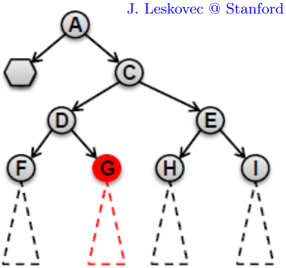


- Shall we continue building the tree?
  - If **Yes**, G is internal, which attribute to split on (test)?
  - If **No**, G is leaf, what is the prediction rule?

Building Decision Tree

Stop when the leaf (subtree at G)

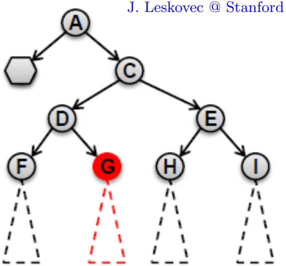
- is pure (purity?) or
- When the size of sub-dataset at G is small e.g. $|D_G| \leq 5$
- ...



Building Decision Tree

If we stop at G, then prediction at G can be

- mode of class labels in sub-dataset D_G
- For a numeric target variable prediction could be an average of target variable values in D_G
- When target variable is numeric it is called Regression Tree

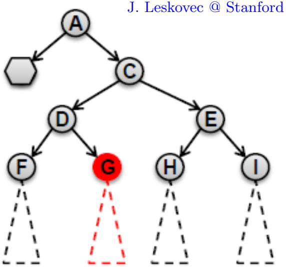


Attribute Selection

Building Decision Tree

Top attributes are selected based on metrics e.g.

- Entropy
- Information Gain
- Gini Index



Common algorithms for Decision Tree are ID3, C4.5, ...

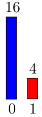
Entropy

In information theory, entropy quantify the average level of information content or uncertainty in a random variable

Flip a fair and a biased coin

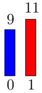
Outcome of Coin 1

1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 0 0 0 $\cdots ?$



Outcome of Coin 2

1 0 0 1 1 0 1 1 0 1 0 1 0 1 0 1 0 1 0 1 $\cdots ?$



In which case would we be more surprised if the next outcome is a 1?

- Entropy values range between 0 and 1 bit ▷ unit of entropy
- Max surprise is for fair coin ($p = 1/2$) ▷ no reason to expect an outcome over another
- Min entropy value is 0 bit for $p = 0$ or $p = 1$

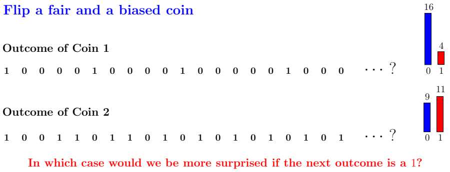
These slides about information theory concepts are adapted from Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Entropy

A random variable X taking value $x_1, \ldots, x_n$ has entropy

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

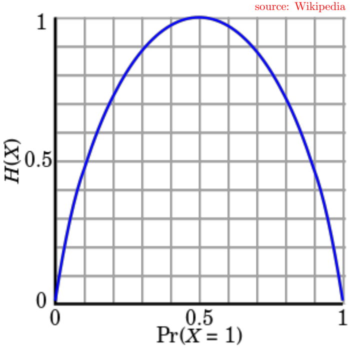
- For fair coin, $p = 1/2$, $\quad H(\cdot) = -1/2 \log 1/2 - 1/2 \log 1/2 = 1$
- For 1-sided coin, $p = 1/0$, $\quad H(\cdot) = -1 \log 1 - 0 \log 0 = 0$
- For coin-1 in example, $\quad H(\cdot) = -16/20 \log 16/20 - 4/20 \log 4/20 = 0.721928$
- For coin-2 in example, $\quad H(\cdot) = -9/20 \log 9/20 - 11/20 \log 11/20 = 0.99277$



Entropy

A random variable X taking value $x_1, \ldots, x_n$ has entropy

$$H(X) \;=\; -\sum_{i=1}^{n} p(x_i)\log p(x_i)$$



Entropy $H(X)$ (expected surprisal) of a coin flip (in bits) plotted versus the bias of the coin $Pr(X = 1) = P(heads)$

Entropy of joint distribution

Entropy of the joint distribution of random variables X and Y

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

$$
\begin{aligned}
H(X,Y) &= -\sum_{x\in X}\sum_{y\in Y} p(x,y)\log_2 p(x,y) \\
&= -\frac{24}{100}\log_2\frac{24}{100} - \frac{1}{100}\log_2\frac{1}{100} - \frac{25}{100}\log_2\frac{25}{100} - \frac{50}{100}\log_2\frac{50}{100} \\
&\approx 1.56\text{bits}
\end{aligned}
$$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Conditional Entropy

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness Y, **given that it is raining**?

$$\begin{aligned} H(Y|X=x) &= -\sum_{y\in Y} p(y|x)\log_2 p(y|x) \\ &= -\frac{24}{25}\log_2\frac{24}{25} - \frac{1}{25}\log_2\frac{1}{25} \\ &\approx 0.24\text{bits} \end{aligned}$$

- We used: $p(y|x) = \frac{p(x,y)}{p(x)}$, and $p(x) = \sum_y p(x,y)$ (sum in a row)

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Conditional Entropy

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- The expected conditional entropy:

$$
\begin{aligned}
H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) \\
&= -\sum_{x \in X} \sum_{y \in Y} p(x,y) \log_2 p(y|x)
\end{aligned}
$$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Conditional Entropy

- Example: $X = \{\text{Raining, Not raining}\}$, $Y = \{\text{Cloudy, Not cloudy}\}$

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- What is the entropy of cloudiness, given the knowledge of whether or not it is raining?

$$\begin{aligned} H(Y|X) &= \sum_{x \in X} p(x) H(Y|X=x) \\ &= \frac{1}{4} H(\text{cloudy}|\text{is raining}) + \frac{3}{4} H(\text{cloudy}|\text{not raining}) \\ &\approx 0.75 \text{ bits} \end{aligned}$$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Conditional Entropy

- Some useful properties:
  - H is always non-negative
  - Chain rule: $H(X, Y) = H(X|Y) + H(Y) = H(Y|X) + H(X)$
  - If X and Y independent, then X doesn't tell us anything about Y: $H(Y|X) = H(Y)$
  - But Y tells us everything about Y: $H(Y|Y) = 0$
  - By knowing X, we can only decrease uncertainty about Y: $H(Y|X) \leq H(Y)$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Information Gain

	Cloudy	Not Cloudy
Raining	24/100	1/100
Not Raining	25/100	50/100

- How much information about cloudiness do we get by discovering whether it is raining?

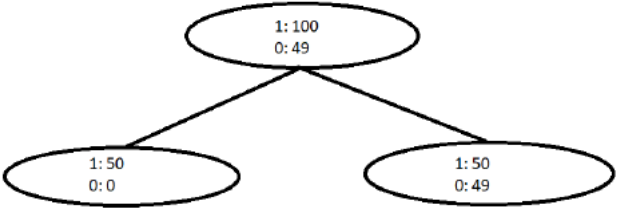
$$\begin{aligned} IG(Y|X) &= H(Y) - H(Y|X) \\ &\approx 0.25 \text{ bits} \end{aligned}$$

- This is called the information gain in Y due to X, or the mutual information of Y and X
- If X is completely uninformative about Y: $IG(Y|X) = 0$
- If X is completely informative about Y: $IG(Y|X) = H(Y)$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Information Gain

- Information gain measures the informativeness of a variable, which is exactly what we desire in a decision tree attribute!
- What is the information gain of this split?



- Root entropy: $H(Y) = -\frac{49}{149}\log_2(\frac{49}{149}) - \frac{100}{149}\log_2(\frac{100}{149}) \approx 0.91$
- Leafs entropy: $H(Y|left) = 0$, $H(Y|right) \approx 1$.
- $IG(split) \approx 0.91 - (\frac{1}{3}\cdot 0 + \frac{2}{3}\cdot 1) \approx 0.24 > 0$

Grosse, Farahmand, & Carrasquilla, Uni. of Toronto

Classification: Some other Concepts

Some other concepts related to classification you should be familiar with

- Decision boundary
- ROC-Curve
- Multi-Class classification form binary classifier
  -
  - One-vs-All
  - One-vs-Rest
- Some classifiers you should read about, (at least wikipedia level is essential for reading papers and using them in your projects)
- Random Forest, Support Vector Machine, Neural Networks, Deep Learning