



Melbourne Bioinformatics

BIOINFORMATICS + DATA SERVICES + INFRASTRUCTURE, FOR LIFE SCIENCES TODAY



COMP90014

Algorithms for Bioinformatics

Week 11A: Supervised Learning

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

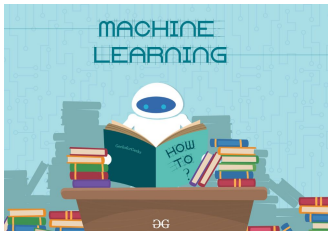
Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

Machine learning



Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.

- Arthur Samuel, 1959.

Machine Learning is the study of algorithms that:

- *Improve their performance P*
- *at some task T*
- *with experience E .*

A well-defined learning task is given by $\langle P, T, E \rangle$

- Tom Mitchell, 1998.

Application of Unsupervised Clustering

GeoGuessr - Online Game

Google maps street view image is shown, player guesses the location.

Closer to the actual location = better score.

Difficult problem for AI

Images taken from anywhere, not uniformly sampled (cities vs rural)

Daytime / nighttime, varying weather, Illumination, season, traffic etc

Predicting latitudes and longitudes directly = poor performance

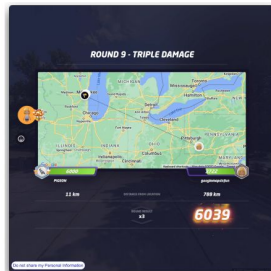
Latitude / longitude not really good classification metric - regression.

Modern approaches use **geocells** to divide space into categories - classification

Unsupervised clustering used to define geocells



(a) Sample image in a Geoguessr location.

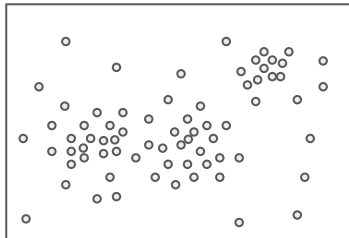


(b) Sample comparison of guesses between PIGEON and a human player.

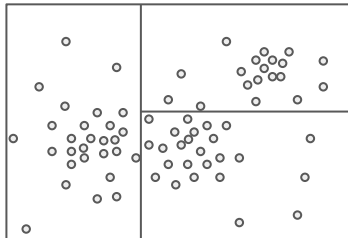
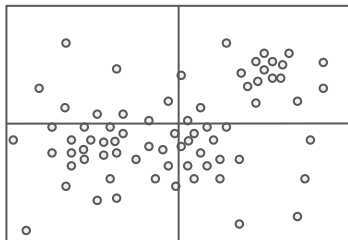
PIGEON: Initial idea for Geocell Creation

这张图片描述了一个名为PIGEON的地理单元创建的初始想法。
PIGEON在这里指的可能是一个数据分析或数据处理的方法或工具

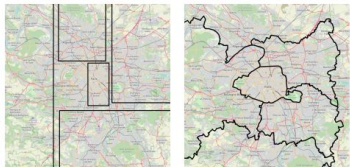
Previous work:
Simple subdivision into
4 quadrants



PIGEON:
Divisive, hierarchical
k=2 k-means clustering



PIGEON: Selected idea for Geocell Creation



(a) With rectangular geocells. (b) With our semantic geocells.



Figure 3. Voronoi tessellation applied in the process of geocell creation.

Algorithm 1 Semantic Geocell Division Algorithm

Input: geocell boundaries g , training samples x , OPTICS parameters p , minimum cell size MINSIZE.
Initialize $j = 1$.
repeat
 Initialize $C \leftarrow \text{OPTICS}(p_j)$.
 for g_i **in** g **do**
 Define $x_i = \{x_j | x_j \in x \wedge x_j \in g_i\}$.
 repeat
 Cluster $c = C(x_i)$.
 $c_{max} = c_k$ where $|x_{i,k}| \geq |x_{i,l}| \forall l$.
 if $|c_{max}| > \text{MINSIZE}$ and $|x \setminus x_{i,k}| > \text{MINSIZE}$
 then
 New cell $g_{new} = \text{VORONOI}(x_{i,k})$.
 $g_i = g_i \setminus g_{new}$.
 Assign x_i to cells i and new .
 end if
 until convergence
 end for
 $j = j + 1$
 until j is $|p|$

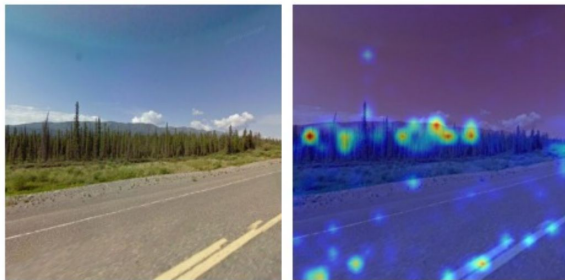
PIGEON: Performance & Quirks

Works really well

On-par or better than human world
champion

Neural net was analysed

Which features of an image was it paying
attention to?



(a) Attention attribution map for an image in Canada.

PIGEON: Performance & Quirks

Works really well

On-par or better than human world
champion

Neural net was analysed

Which features of an image was it paying
attention to?



(a) Attention attribution map for an image in Canada.

PIGEON: Performance & Quirks

Works really well

On-par or better than human world champion

Neural net was analysed

Which features of an image was it paying **attention** to?



(a) Attention attribution map for an image in Canada.

PIGEON worked out an interesting feature to use - the car!

Has time and location data for each image.

Knows that images taken close in time & close in location were done by same car

Could use features of the car / camera to identify a new image (eg smudges on the lens)

How do we feel about this?
Perhaps... **overfitting?**

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

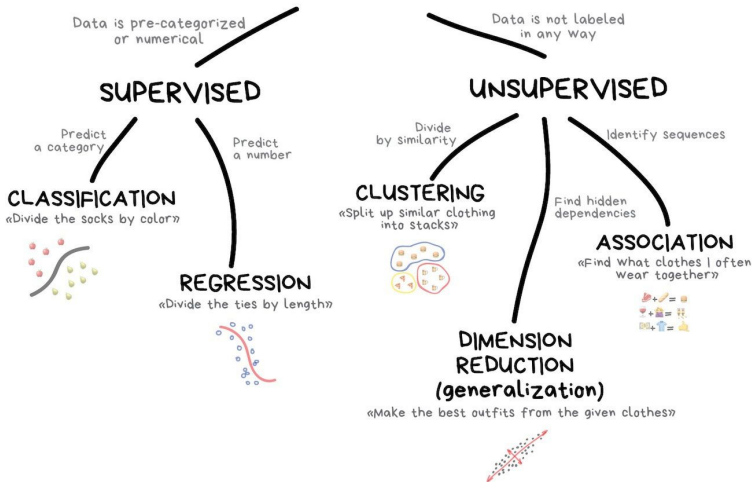
Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

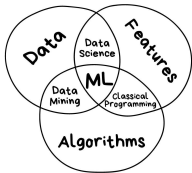
CLASSICAL MACHINE LEARNING



Three components of supervised learning

Data

- 👑 labelled (historical / experimental)
- 👑 representative and diverse
 - collection and curation is key
 - **A predictive model is as good as the training data**



Features

- 👑 properties to be used as evidence to train a predictive model
- 👑 requires knowledge of the problem

Learning Algorithms

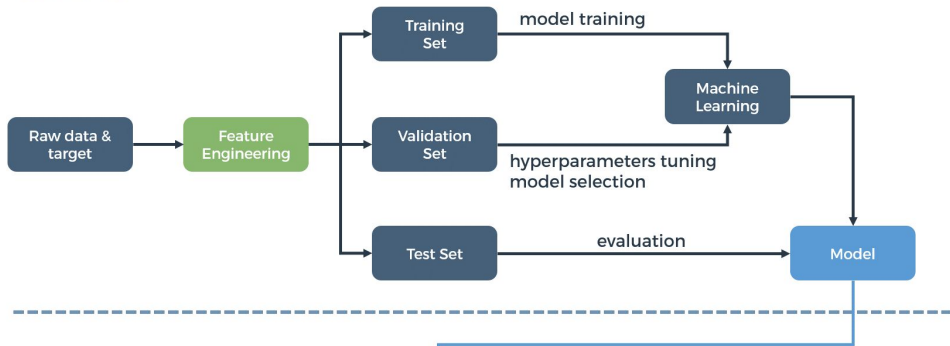
- 👑 which one?

Infer a function f :

- 👑 maps input (features) to an output (target)
- 👑 experience (examples of input-output pairs)

Supervised Learning Pipeline

TRAINING



PREDICTING



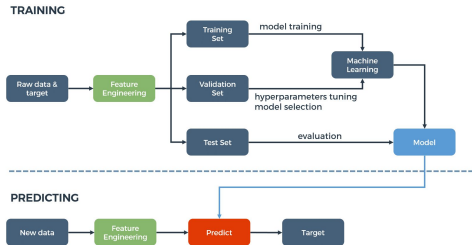
Data

Training set

- ☛ representative set of examples used for training, where the target value is known

Validation set

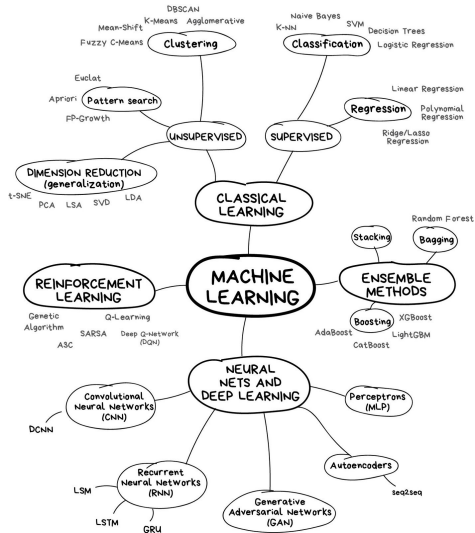
- ☛ representative set of examples used to tune the architecture of a learning algorithm and estimate prediction errors



Independent test set (blind test)

- ☛ independently assess the performance of a predictive model
- ☛ never used during the training process
- ☛ error on the blind test provides an unbiased estimate of the generalization error

Learning algorithms



There are tons of algorithms

- 🏠 K-nearest neighbours (KNN)
- 🏠 Naïve Bayes
- 🏠 Support Vector Machines (SVMs)
- 🏠 Decision Trees
- 🏠 not an exhaustive list

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

Association Rule Learning

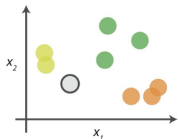
Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

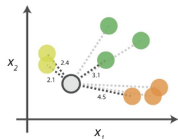
K-nearest neighbours (KNN)

0. Look at the data











Say you want to classify the grey point into a class. Here, there are three potential classes - lime green, green and orange.

1. Calculate distances









Start by calculating the distances between the grey point and all other points.

2. Find neighbours

Point	Distance	
 	2.1	→ 1st NN
 	2.4	→ 2nd NN
 	3.1	→ 3rd NN
 	4.5	→ 4th NN

Next, find the nearest neighbours by ranking points by increasing distance. The nearest neighbours (NNs) of the grey point are the ones closest in dataspace.

3. Vote on labels

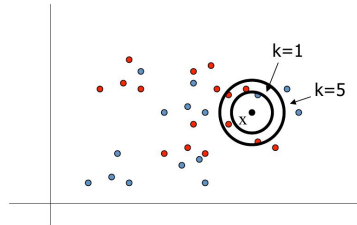
Class	# of votes	
	2	→ Class  wins the vote! Point  is therefore predicted to be of class  .
	1	
	1	

Vote on the predicted class labels based on the classes of the k nearest neighbours. Here, the labels were predicted based on the $k=3$ nearest neighbours.

- given training data, $D = \{(x_1, y_1), \dots, (x_N, y_N)\}$ and a test point x_u
- prediction rule: look at the K most similar training examples to x_u
- for classification: assign the majority class label (majority voting)
- for regression: assign the average response
- The algorithm requires
 - parameter K : number of nearest neighbors to look for
 - distance function: to compute similarities between points

KNN algorithm

```
1 Procedure KNN:
2   foreach test point do
3     Compute the distance to each training point
4     Sort the distances in ascending order
5     Select the  $K$ -nearest neighbors
6     if classification then use majority rule
7     if regression then use averaging
```



- 🧠 KNN is called a non-parametric method
- 🧠 Unlike other supervised learning algorithms, it doesn't learn an explicit mapping function f from the training data (no explicit model)
- 🧠 It simply uses the training data at the test time to make predictions

KNN implementation

```
1  # Locate the most similar neighbors
2  def get_neighbors(train, test_row, num_neighbors):
3      distances = list()
4      for train_row in train:
5          dist = euclidean_distance(test_row, train_row)
6          distances.append((train_row, dist))
7      distances.sort(key=lambda tup: tup[1])
8      neighbors = list()
9      for i in range(num_neighbors):
10         neighbors.append(distances[i][0])
11     return neighbors
12
13  # Make a prediction with neighbors
14  def predict_classification(train, test_row, num_neighbors):
15      neighbors = get_neighbors(train, test_row, num_neighbors)
16      output_values = [row[-1] for row in neighbors]
17      prediction = max(set(output_values), key=output_values.count)
18      return prediction
```

KNN implementation

```
1 # kNN Algorithm
2 def k_nearest_neighbors(train, test, num_neighbors):
3     predictions = list()
4     for row in test:
5         output = predict_classification(train, row, num_neighbors)
6         predictions.append(output)
7     return(predictions)
```

- ⌚ Time complexity?
- ⌚ Space complexity?
- ⌚ Would KNN be adequate for large-scale, real-time predictions?

Properties of KNN

$$d(x_i, x_j) = \sum_{m=1}^D (x_{im} - x_{jm})^2$$

KNN requires computing distances

- ☞ several different distance metrics (e.g. Euclidean)

Features should be on the same scale

- ☞ e.g. kcal and kJ (we need to normalize them)

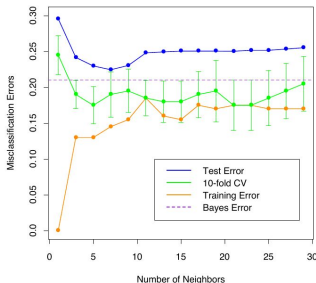
Choosing K

Pros:

- ☞ simple and intuitive, easily to implement/interpret

Cons:

- ☞ memory usage (store training data in memory)
- ☞ may perform badly in high dimensions
- ☞ sensitive to noisy features
- ☞ categorical attributes?



Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

Naïve Bayes

Conditional probability:

$$P(A | B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A and B

Probability of A given B

Probability of B

LIKELIHOOD
the probability of "B"
being TRUE given that "A" is TRUE

PRIOR
the probability of
"A" being TRUE

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

POSTERIOR
the probability of "A"
being TRUE given that "B" is TRUE

The probability
of "B" being
TRUE

@luminousmen.com

Maximum a posteriori (MAP):

$\arg \max P(A | B)$ for each class A

$$\arg \max P(A | B) = \arg \max P(B | A) \times \frac{P(A)}{P(B)}$$
$$= \arg \max P(B | A) \times P(A)$$

- uses Bayes's rule
- $P(B)$ will be the same for all classes
- for a set of features x_i we calculate the joint probability:
 $B = (x_1, x_2, \dots, x_n)$
- $\arg \max P(x_1, x_2, \dots, x_n | A) \times P(A)$
 $P(x_1, x_2, \dots, x_n | A) = \prod_i P(x_i | A)$
- simply the product of individual probabilities
- assumption: features are independent given a class

Naïve Bayes Algorithm

Cough	Sneezing	Fever	Flu
Yes	Yes	No	No
Yes	No	Yes	Yes
No	No	No	No
No	Yes	Yes	Yes

- Given this training data, if we have a new patient with only a cough, should we diagnose them with the flu?
- $P(? \mid \text{Cough, No Sneezing, No Fever})$
- $P(\text{Cough, No Sneezing, No Fever} \mid \text{Flu}) \times P(\text{Flu})?$
- $P(\text{Cough, No Sneezing, No Fever} \mid \text{No Flu}) \times P(\text{No Flu})?$

```
1 Procedure Naïve Bayes Learn(examples):  
2   foreach target value  $v_j$  do  
3      $\hat{P}(V_j) \leftarrow \text{estimate } P(V_j)$   
4     foreach attribute value  $a_i$  of each attribute  $a$   
5       do  
6          $\hat{P}(a_i \mid v_j) \leftarrow \text{estimate } P(a_i \mid v_j)$ 
```

```
1 Procedure Classify New Instance( $x$ ):  
2    $v_{NB} = \arg \max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i \mid v_j)$ 
```


Naïve Bayes Algorithm

```
1 Procedure Naïve Bayes Learn(examples):  
2   foreach target value  $v_j$  do  
3      $\hat{P}(V_j) \leftarrow$  estimate  $P(V_j)$   
4     foreach attribute value  $a_i$  of each attribute  $a$   
5       do  
6          $\hat{P}(a_i | v_j) \leftarrow$  estimate  $P(a_i | v_j)$ 
```

```
1 Procedure Classify New Instance( $x$ ):  
2    $v_{NB} = \arg \max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i | v_j)$ 
```

Type	Long	Not long	Sweet	Not sweet	Yellow	Not yellow	Total
Banana	400	100	350	150	450	50	500
Orange	0	300	150	150	300	0	300
Other	100	100	150	50	50	150	200
Total	500	500	650	350	800	200	1000

Step 1: Compute the 'Prior' probabilities for each of the classes.

$$\text{👑 } P(\text{Banana}) = \frac{500}{1000} = 0.50$$

$$\text{👑 } P(\text{Orange}) = \frac{300}{1000} = 0.30$$

$$\text{👑 } P(\text{Other}) = \frac{200}{1000} = 0.20$$

Naïve Bayes Algorithm

```

1 Procedure Naïve Bayes Learn(examples):
2   foreach target value  $v_j$  do
3      $\hat{P}(V_j) \leftarrow$  estimate  $P(V_j)$ 
4     foreach attribute value  $a_i$  of each attribute  $a$ 
5       do
6          $\hat{P}(a_i | v_j) \leftarrow$  estimate  $P(a_i | v_j)$ 

```

```

1 Procedure Classify New Instance(x):
2    $v_{NB} = \arg \max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i | v_j)$ 

```

Type	Long	Not long	Sweet	Not sweet	Yellow	Not yellow	Total
Banana	400	100	350	150	450	50	500
Orange	0	300	150	150	300	0	300
Other	100	100	150	50	50	150	200
Total	500	500	650	350	800	200	1000

Step 2: Compute the 'Likelihood' probabilities for each feature/class combination.

$$\text{👑 } P(\text{Long} \mid \text{Banana}) = \frac{400}{500} = 0.80$$

$$\text{👑 } P(\text{Sweet} \mid \text{Banana}) = \frac{350}{500} = 0.70$$

$$\text{👑 } P(\text{Yellow} \mid \text{Banana}) = \frac{450}{500} = 0.90$$

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A given B Probability of B

Naïve Bayes Algorithm

```
1 Procedure Naïve Bayes Learn(examples):  
2   foreach target value  $v_j$  do  
3      $\hat{P}(V_j) \leftarrow$  estimate  $P(V_j)$   
4     foreach attribute value  $a_i$  of each attribute  $a$   
5       do  
6          $\hat{P}(a_i | v_j) \leftarrow$  estimate  $P(a_i | v_j)$ 
```

```
1 Procedure Classify New Instance(x):  
2    $v_{NB} = \arg \max_{v_j \in V} \hat{P}(v_j) \prod_{a_i \in x} \hat{P}(a_i | v_j)$ 
```



Type	Long	Not long	Sweet	Not sweet	Yellow	Not yellow	Total
Banana	400	100	350	150	450	50	500
Orange	0	300	150	150	300	0	300
Other	100	100	150	50	50	150	200
Total	500	500	650	350	800	200	1000

Step 3: Compute the 'Posterior' probabilities for each class given the features.

$P(Y | \text{Long, Sweet, Yellow})?$

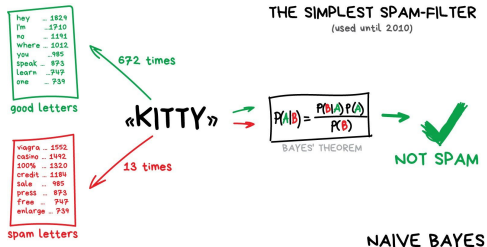
$$P(\text{Banana} | \text{Long, Sweet, Yellow}) \\ = (0.8 \times 0.7 \times 0.9) \times 0.5 = 0.252$$

$$P(\text{Orange} | \text{Long, Sweet, Yellow}) \\ = \dots = 0.0 \rightarrow P(\text{Long} | \text{Orange}) = 0$$

$$P(\text{Other} | \text{Long, Sweet, Yellow}) \\ = \dots = 0.019$$



Naïve Bayes



Pros

- 👑 Easy to implement
- 👑 Very efficient

Cons

- 👑 Independence assumption doesn't always hold
- 👑 Loss of accuracy

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

Association Rule Learning

Tries to find “interesting” **dependent** associations between variables.

Cough	Sneezing	Fever	Flu
Yes	Yes	No	No
Yes	No	Yes	Yes
No	No	No	No
No	Yes	Yes	Yes

Association Rule Learning

Tries to find “interesting” **dependent** associations between variables.

Transaction: $X \Rightarrow Y$ (if X then Y)

- {Sneezing} \Rightarrow {Flu}
- {Cough} \Rightarrow {Flu}
- {Sneezing, Cough} \Rightarrow {Flu}

Cough	Sneezing	Fever	Flu
Yes	Yes	No	No
Yes	No	Yes	Yes
No	No	No	No
No	Yes	Yes	Yes

Association Rule Learning

Tries to find “interesting” **dependent** associations between variables.

Transaction: $X \Rightarrow Y$ (if X then Y)

- {Sneezing} \Rightarrow {Flu}
- {Cough} \Rightarrow {Flu}
- {Sneezing, Cough} \Rightarrow {Flu}

Cough	Sneezing	Fever	Flu
Yes	Yes	No	No
Yes	No	Yes	Yes
No	No	No	No
No	Yes	Yes	Yes

Apply metrics known as **rules** on transactions to identify interesting relationships.

Association Rule Learning

Tries to find “interesting” **dependent** associations between variables.

Transaction: $X \Rightarrow Y$ (if X then Y)

- {Sneezing} \Rightarrow {Flu}
- {Cough} \Rightarrow {Flu}
- {Sneezing, Cough} \Rightarrow {Flu}









































Cough	Sneezing	Fever	Flu
Yes	Yes	No	No
Yes	No	Yes	Yes
No	No	No	No
No	Yes	Yes	Yes

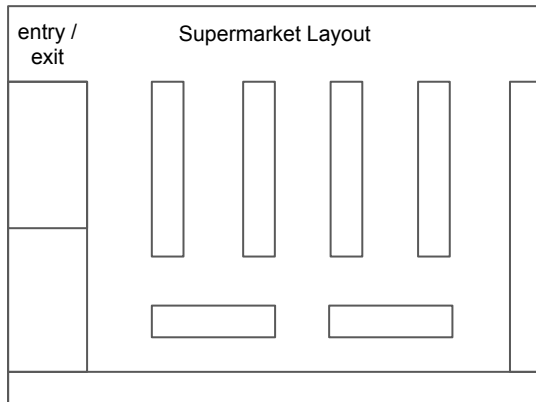
Apply metrics known as **rules** on transactions to identify interesting relationships.

Rule	Description
Support	How frequently does “X” appear in the dataset
Confidence	How many itemsets with “X” also contain “Y”
Lift	Ratio of observed support, to expected support if “X” and “Y” were independent
Conviction	How likely an association is genuine, not just random chance

Association Rule Learning









































Application: Product placement in supermarkets

Transactions	
    	   
   	   
   	 
  	   
    	    

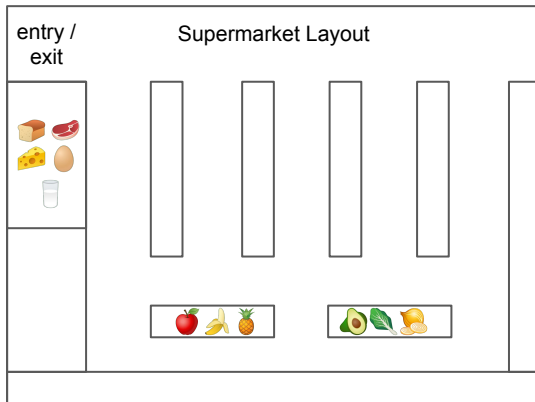


Association Rule Learning

Application: Product placement in supermarkets









































Transactions	
    	   
   	   
   	 
  	   
    	    

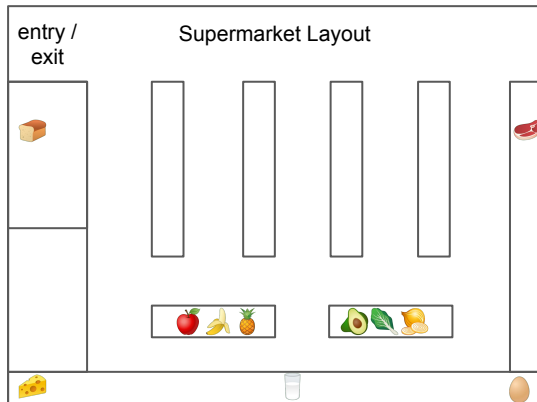
Easiest for customer



Association Rule Learning









































Application: Product placement in supermarkets

Transactions	
    	   
   	   
   	 
  	   
    	    

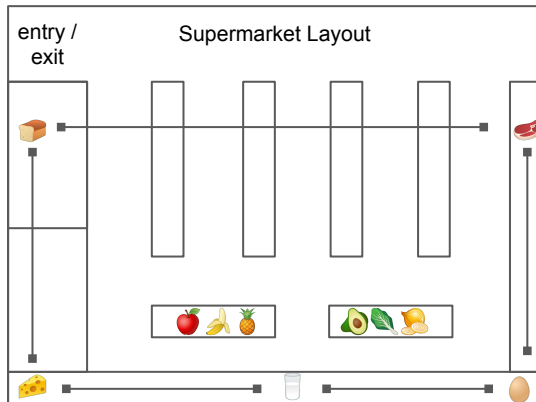


Association Rule Learning

Application: Product placement in supermarkets









































Transactions	
    	   
   	   
   	 
  	   
    	    

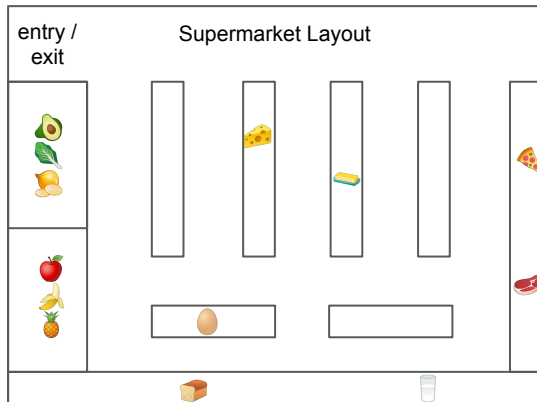
Max distance



Association Rule Learning









































Application: Product placement in supermarkets

Transactions	
    	   
   	   
   	 
  	   
    	    

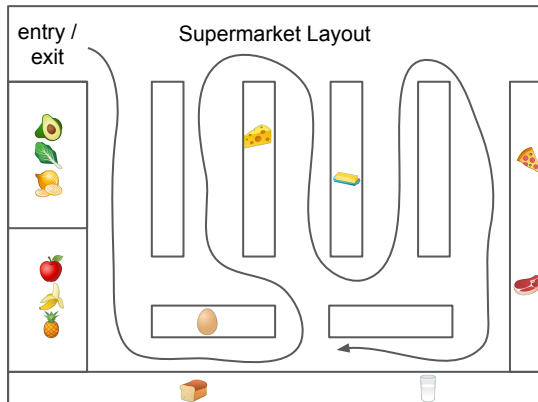


Association Rule Learning

Application: Product placement in supermarkets

Transactions	
    	   
   	   
   	 
  	   
    	    

Maximum walk



Association Rule Learning

Example: [Apriori algorithm](#)

Bottom up approach where subsets
extended by 1 item at a time

Uses BFS to explore item sets efficiently

Identifies branches which will not meet
minimum cutoff criteria

Cutoff Criteria
Min Support = 3

Transactions

Itemsets
{1,2,3,4}
{1,2,4}
{1,2}
{2,3,4}
{2,3}
{3,4}
{2,4}

1 item

Item	Support
{1}	3
{2}	6
{3}	4
{4}	5

2 items

Item	Support
{1,2}	3
{1,3}	1
{1,4}	2
{2,3}	3
{2,4}	4
{3,4}	3

3 items

Item	Support
{2,3,4}	2

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

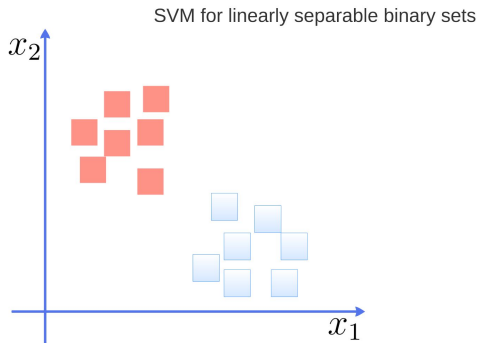
Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

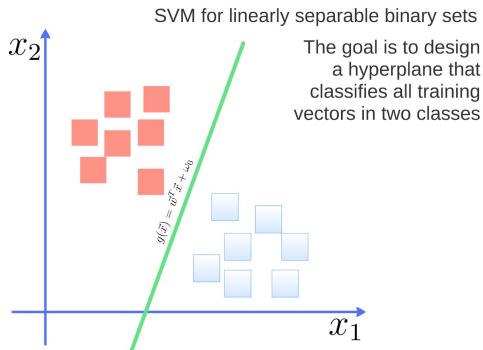
Ensemble Methods

Support Vector Machines (SVM)



- consider linearly separable points
- for given training data, find decision boundaries separating classes (**hyperplane**)
- maximises the perpendicular distance to the nearest points
- **maximum-margin hyperplane**
- gives a linear separator
- optimisation problem

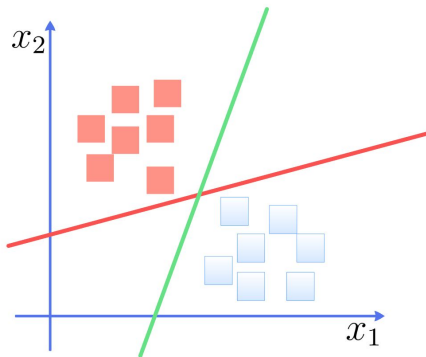
Support Vector Machines (SVM)



- consider linearly separable points
- for given training data, find decision boundaries separating classes (**hyperplane**)
- maximises the perpendicular distance to the nearest points
- maximum-margin hyperplane**
- gives a linear separator
- optimisation problem

Support Vector Machines (SVM)

The best choice will be the hyperplane that leaves the maximum margin from both classes

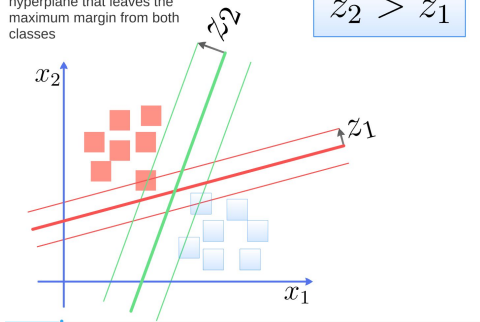


- consider linearly separable points
- for given training data, find decision boundaries separating classes (**hyperplane**)
- maximises the perpendicular distance to the nearest points
- **maximum-margin hyperplane**
- gives a linear separator
- optimisation problem

Support Vector Machines (SVM)

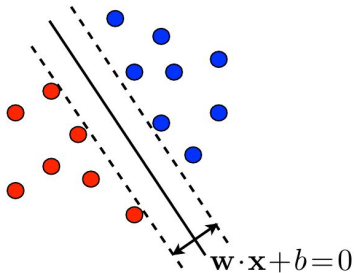
The best choice will be the hyperplane that leaves the maximum margin from both classes

$$z_2 > z_1$$



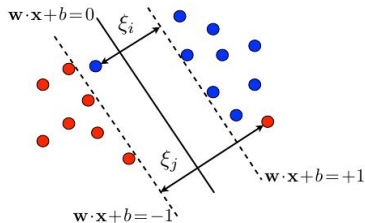
- consider linearly separable points
- for given training data, find decision boundaries separating classes (**hyperplane**)
- maximises the perpendicular distance to the nearest points
- maximum-margin hyperplane**
- gives a linear separator
- optimisation problem

Support Vector Machines (SVM)

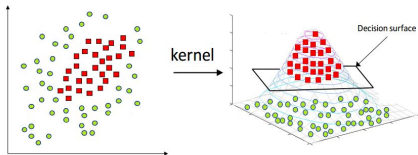


- finding maximum-margin hyperplane is a quadratic optimisation problem to find vector weights w and b
- w can be found as a linear combination of a subset of the training points
 - Points along the margins - the support vectors
- computationally straightforward
- minimising a convex function will find global optimum solution
- gives a linear boundary with maximum distance to points

What if our data is not linearly separable?



- 👑 Use a soft SVM
 - Add terms to objective function to penalise points on the “wrong side” of boundary, but don’t forbid them
- 👑 Use a kernel function
 - Transform input data into a separable space
 - Kernel SVM allows non-linear boundaries in the original space and is a popular approach



Polynomials of degree exactly d : $K(u, v) = (u \cdot v)^d$

Polynomials of degree up to d : $K(u, v) = (u \cdot v + 1)^d$

Gaussian kernels: $K(\vec{u}, \vec{v}) = \exp\left(-\frac{\|\vec{u} - \vec{v}\|_2^2}{2\sigma^2}\right)$

Sigmoid: $K(u, v) = \tanh(\eta u \cdot v + \nu)$

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

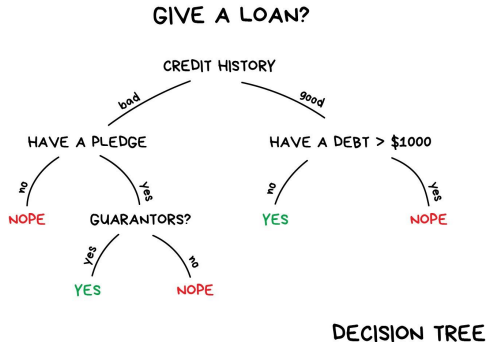
Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

Ensemble Methods

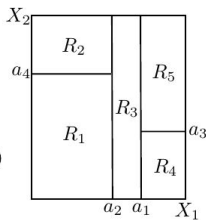
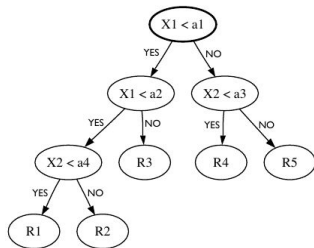
Decision Trees



- a decision tree consists of a hierarchy of rules for prediction
- we can visualise this as a tree where every node is either:
 - an internal node, with a decision criterion, with two children (binary)
 - a leaf node, which predicts an outcome
- trees can be built for classification or for regression

How to build a decision tree from data?

- 1 Choose a decision point yielding best purity
- 2 Partition data into corresponding subsets
- 3 Reiterate with resulting subsets
- 4 Stop when regions are approximately pure



Impurity in classification

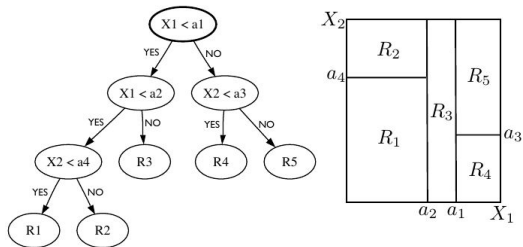
- 👑 misclassification
- 👑 Gini impurity: probability of incorrectly classifying a randomly chosen data point

Impurity in regression

- 👑 mean squared error

$$F(R) = \sum_{x_i \in R} (y_i - \langle y \rangle)^2$$

Recursive binary splitting is a greedy heuristic



Prediction

- a path from root to leaf based on the features

Pros

- decision trees facilitate explaining complex data
 - interpretable
- results can be easy to analyze and understand
- different types of variables: categorical, numerical

Cons

- large trees can easily overfit
- use a pruning criteria
 - minimize (impurity + tree size)

Supervised Learning

Recap & brief detour about GeoGuessr

Supervised learning

K-Nearest Neighbors (KNN)

Naive Bayes

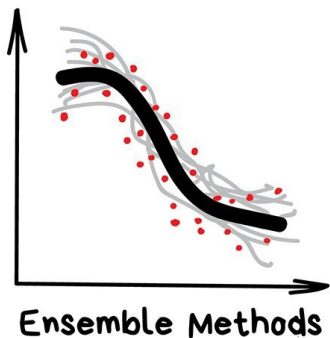
Association Rule Learning

Support Vector Machines (SVM)

Decision Trees

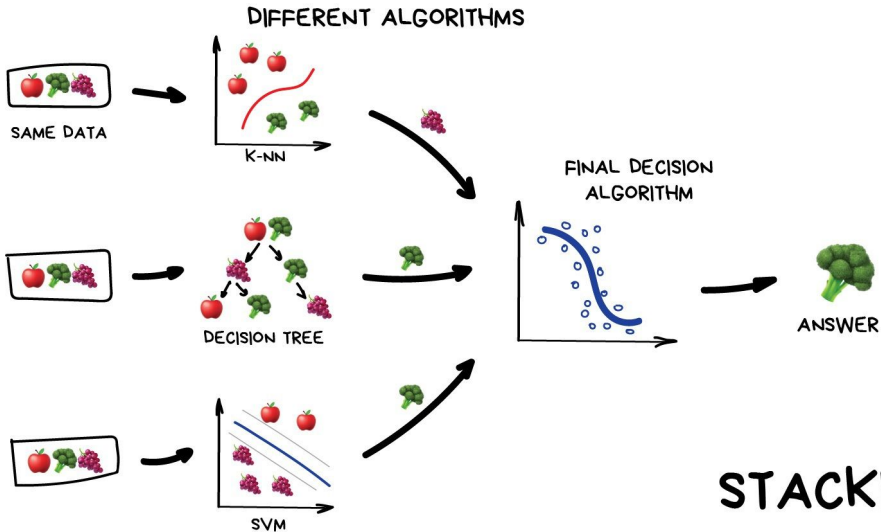
Ensemble Methods

Ensemble methods

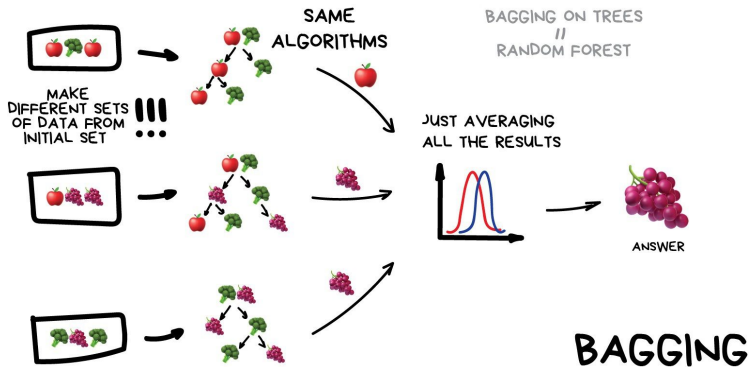


- rather than just one, combine multiple learning algorithms (or instances) into one predictive model
- ensemble methods are meta-algorithms aiming to:
 - decrease variance;
 - decrease bias;
 - improve predictions
- Stacking
- Bagging
- Boosting

Stacking



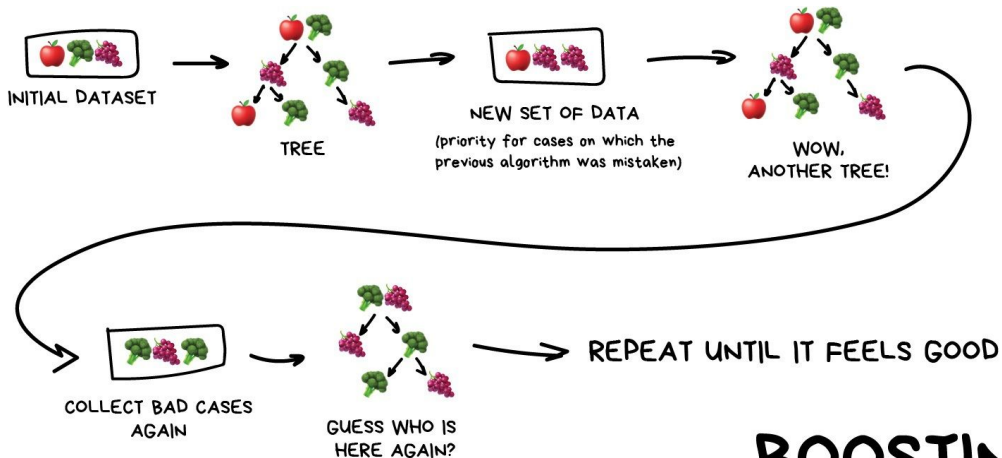
Bagging



Random Forest ([Breiman et al., 2001](#))

- 👤 Very popular and effective
- 👤 Classification & regression
- 👤 Robust to outliers & feature importance

Boosting



BOOSTING

Thank you!

Today: Supervised Learning

Next time: Model selection, tuning, validation