

انهارده هنتكلم علي K nearest neighbor و ال model selection .. الموديل سيليكشن هنا هنتكلم شويه برضو علي ال decision trees .. الدكتور
سأل سؤال ..

Q&A

Q: Why don't my entropy calculations match those on the slides?

A: $H(Y)$ is conventionally reported in "bits" and computed using log base 2.
e.g., $H(Y) = -P(Y=0) \log_2 P(Y=0) - P(Y=1) \log_2 P(Y=1)$

Q: Why is entropy based on a sum of $p(\cdot) \log p(\cdot)$ terms?

A: We don't have time for a full treatment of why it has to be this, but we can develop the right intuition with a few examples...

أول سؤال في السلايد ديه .. ليه ال entropy calculations مش زي اللي الدكتور بيحسبها في السلايد .. الدكتور رد وقال .. انو قال قبل كذا ان ال entropy بنستخدمو و ن report اللي بيطلع علي هيئة bits باستخدام ال log base 2 .. بس لو انت بدل log2 خليتها log10 .. انت هت report in Nats .. فاستخدم ال logbase2 ..

تاني سؤال .. ليه بضرب الاحتمالية في اللوج بتاعها؟ .. الاجابه مفيش وقت ليها ..

Q&A

Q: How do we deal with ties in k-Nearest Neighbors (e.g. even k or equidistant points)?

A: I would ask you all for a good solution!

Q: How do we define a distance function when the features are categorical (e.g. weather takes values {sunny, rainy, overcast})?

A: Step 1: Convert from categorical attributes to numeric features (e.g. binary)
Step 2: Select an appropriate distance function (e.g. Hamming distance)

سؤال تاني .. ازاي نهاندل ال tie في ال k-Nearest neighbor .. قال انو احنا اللي هنجابو عليه ... هه

سؤال رابع .. ازاي نعرّف ال distance between neighbors ؟ .. في الكلاس كنا بعينا كذا بنقول ده اقرب .. هتعمل ايه لو كانت ال attributes categorical .. فانت هتحول ال categorical attributes ل numeric features .. اختار بعد كذا distance function تبقا مناسبه للي انت عاوزو ..

يلا بيبدأ ننتكلم علي ال overfitting ...

Decision Tree Generalization

Question:

Which of the following would generalize best to unseen examples?

- A. Small tree with low training accuracy
- B. Large tree with low training accuracy
- C. Small tree with high training accuracy
- D. Large tree with high training accuracy

Answer:



11

تعال بقي نخط شوية terms تساعدنا reason more علي اللي بيحصلنا واحنا شغالين .. هنبدأ بال overfitting .. هنا بيحصل لما الموديل بيقتا too complex فبذل ما بيعمل fitting to the underlying trend in the data .. تلاقيه بيعمل fitting to the noise in the data ففعلياً ممكن متيقاش النويز بتاعت الداتا انما ممكن نبقا random statistical fluctuations .. فانت ممكن مييقاش عندك inductive bias يوديها ناحية ال generalization .. مثال علي كذا ال ... memorizer algorithm ميديكال ستودنت عمال يحفظ ال case studies انما مش عارف يطبقها ازاي ... ال Underfitting هو موديل بسيط جداً مش قادر ي capture the trend in the data .. فهو overly biased ناحية حاجه بسيطه .. حاجه مثلاً زي ال majority vote classifier .. ده ابسط حاجه يعني .. depth 0 decision tree .. حاجه برضو زي ال toddler مدخلش طب .. وقام يشتغل في ال medical diagnosis .. طبعاً حاجه سيئه جداً ..

Overfitting and Underfitting

Underfitting

- The model...
 - is too simple
 - is unable captures the trends in the data
 - exhibits too much bias
- Example: majority-vote classifier (i.e. depth-zero decision tree)
- Example: a toddler (that has **not** attended medical school) attempting to carry out medical diagnosis

Overfitting

- The model...
 - is too complex
 - is fitting the noise in the data
 - or fitting random statistical fluctuations inherent in the “sample” of training data
 - does not have enough bias
- Example: our “memorizer” algorithm responding to an “orange shirt” attribute
- Example: medical student who simply memorizes patient case studies, but does not understand how to apply knowledge to new patients

12

لو احنا قلنا ان عندنا H hypothesis .. احنا عندنا 2 ديفينيشنز ...

Overfitting

- Consider a hypothesis h its...
 - ... error rate over all training data: $\text{error}(h, D_{\text{train}})$
 - ... error rate over all test data: $\text{error}(h, D_{\text{test}})$
 - ... true error over all data: $\text{error}_{\text{true}}(h)$
- We say h overfits the training data if...
 - $\text{error}_{\text{true}}(h) > \text{error}(h, D_{\text{train}})$
- Amount of overfitting =
 - $\text{error}_{\text{true}}(h) - \text{error}(h, D_{\text{train}})$

In practice,
 $\text{error}_{\text{true}}(h)$ is
unknown

Slide adapted from Tom Mitchell

14

هنحط تعريف ثالث وهو ال true error over all data .. تخيل ان عندك حاجه D_{all} special data set .. فانت هتقول $\text{error}(h, D_{\text{all}})$.. بس ال key property هنا انت مش عارف القيمة بكام .. عشان علي ارض الواقع انت متقدرش تعرف كل الداتا .. متقدرش تعرف كل المرضي اللي احنا شخصناهم او هنتشخصهم .. فتعريف ال overfitting هنا هو ان ال h هت overfits the training data لو ال true error كان أكثر من ال training error .. قيمة ال overfitting هو ال training error - true error .. فاحنا هناخد التعريف ده و نحطو في ال practice .. ده عشان في ارض الواقع انت مبتعرفش تحسب مين أعلي من مين ... فبالتالي احنا بنقول علي ال test error هو ال true error .. او بمعنى اصح هو ال approximation بتاع ال true error ..

Overfitting in Decision Tree Learning

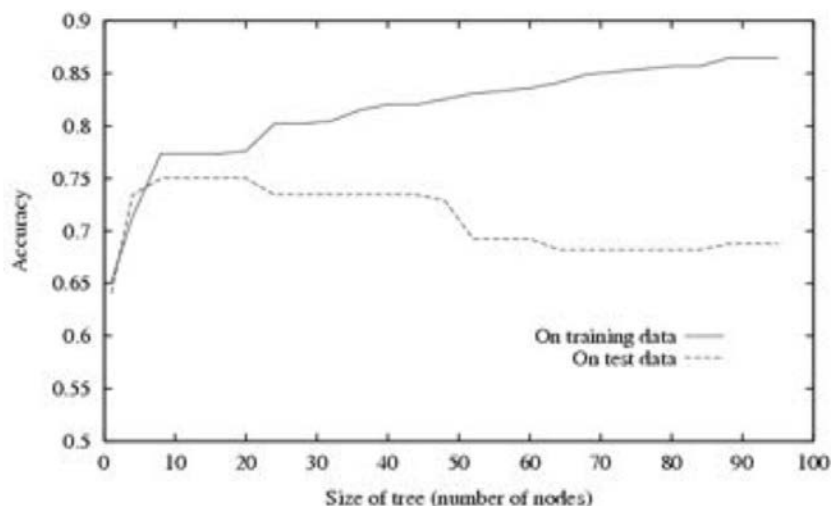


Figure from Tom Mitchell

اللي بنلاحظ في الصورة هنا ان كل ما الحجم بتاع الشجرة زاد .. التريننج إيرور عمال يقل .. انما التست إيرور .. زاد فجاء في الأول .. وبعدين عمال يقل يقل يقل .. ازاي نقدر نحل مشكلة ال overfitting .. في حل انك متزودش أوي في العمق بتاع الشجرة .. ثاني طريقه ان ال splitting criterion مثلا بيقل

ليه threshold معين تحتيه انت متقدرش تعمل ال splitting .. ثالث حل انك مثلاً عندك statistically significant test .. حل رابع .. اعمل شجره عميقه أوي وبعدين اقطع الشجره .. prune it back

How to Avoid Overfitting?

For Decision Trees...

1. Do not grow tree beyond some **maximum depth**
2. Do not split if splitting criterion (e.g. mutual information) is **below some threshold**
3. Stop growing when the split is **not statistically significant**
4. Grow the entire tree, then **prune**

17

هنشوف ازاي حاجه زي كذا ممكن تحصل .. السلايد الجايه بتعبر عن وصف سريع كذا لل reduced error pruning .. الفكره هنا خذ الداتا قسمها ل train-valid-test .. اعمل شجره بت classify the training set perfectly .. وبعدها تعمل ال pruning process لحد ما ال validation accuracy تبدأ تتأثر باللي بيحصل .. فايه الي بيحصل .. انت بت evaluate the impact on the validation set .. انك ت prune كل node وال descendant nodes اللي تحتها .. وبعدين انت greedily remove الواحده اللي فعلاً بتحسّن ال validation accuracy .. ده هيجيبلك أحسن subtree من ال whole tree .. تعال نشوف ايه اللي بيحصل علي الجراف:

Reduced-Error Pruning

Split data into *training* and *validation* set

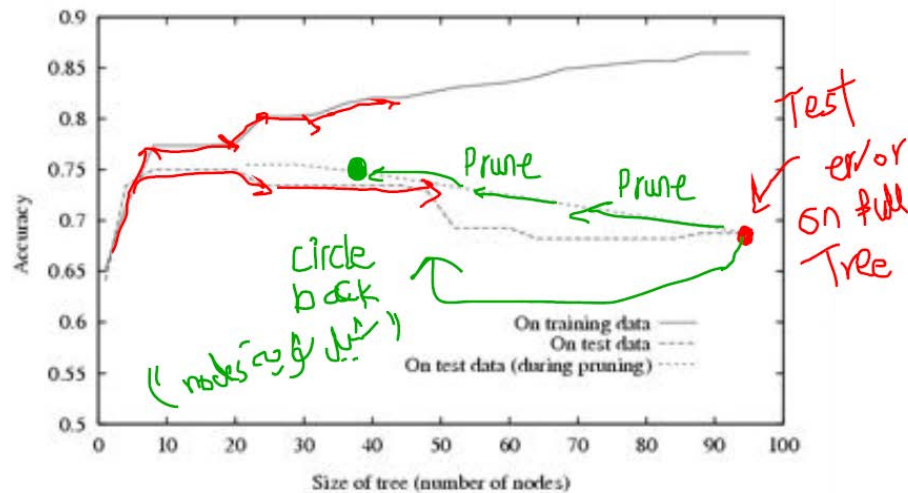
Create tree that classifies *training* set correctly

Do until further pruning is harmful:

1. Evaluate impact on *validation* set of pruning each possible node (plus those below it)
2. Greedily remove the one that most improves *validation* set accuracy

- produces smallest version of most accurate subtree
- What if data is limited?

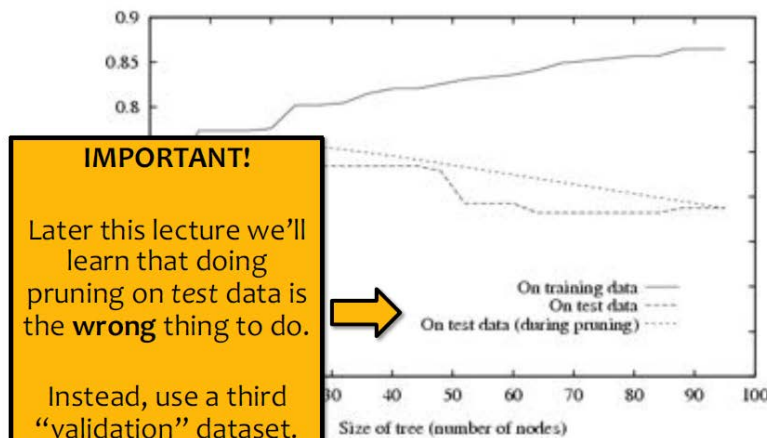
Effect of Reduced-Error Pruning



Slide from Tom Mitchell

19

Effect of Reduced-Error Pruning



Slide from Tom Mitchell

20

هنتكلم شويه برضو علي ال decision trees ... هي من أكثر ال classifications في ارض الواقع .. عشان انت لو قلت مثلاً رحت ال industry وقلت عاوز ابني ال medical diagnosis system .. فتقوم قايل لنفسك .. بس انا سمعت عن ال deep learning .. فهبني اصعب نتورك وارخم نتورك و تقوم جايب اعلي accuracy .. وتقوم باعت للمدير بتاعك انك بنيت الموديل و شغلته و جيت الاكيورسي الجامده جداً ديه .. فالمدير يسألك .. ازاى بتشتغل ؟ .. تقولو انا مش عارف .. محدش عارف .. فترجع ثاني .. تبني decision tree .. الاكيورسي بتاعتها صغيره .. وتقوم باعتها للمدير بتاعك .. تقولو خد الشجره ديه .. وتشرحلو بقا ايه اللي بيحصل .. اول حاجه احنا هنبص علي ال attribute ديه وبعدين نقول هنخش يمين ولا هنخش شمال .. وهكذا .. فالمدير يقولك ديه حاجه بسيطه .. أينعم الاكيورسي أقل .. بس خرينا نشغل بديه .. هي كمان سهله في ال memory .. ويمكن تستخدمها في variety of problems . اقرا السلايد الجايه ..

Decision Trees (DTs) in the Wild

- DTs are one of the most popular classification methods for practical applications
 - Reason #1: The learned representation is **easy to explain** a non-ML person
 - Reason #2: They are **efficient** in both computation and memory
- DTs can be applied to a wide variety of problems including **classification, regression, density estimation**, etc.
- **Applications of DTs** include...
 - medicine, molecular biology, text classification, manufacturing, astronomy, agriculture, and many others
- **Decision Forests** learn many DTs from random subsets of features; the result is a very powerful example of an **ensemble method** (discussed later in the course)

23

من مميزات الألوورزم ده انو ممكن تعاملو معاملة ال Building blocks .. تبني بقا decision forests ..

DT Learning Objectives

You should be able to...

1. Implement Decision Tree training and prediction
2. Use effective splitting criteria for Decision Trees and be able to define entropy, conditional entropy, and mutual information / information gain
3. Explain the difference between memorization and generalization [C1ML]
4. Describe the inductive bias of a decision tree
5. Formalize a learning problem by identifying the input space, output space, hypothesis space, and target function
6. Explain the difference between true error and training error
7. Judge whether a decision tree is "underfitting" or "overfitting"
8. Implement a pruning or early stopping method to combat overfitting in Decision Tree learning

24

دلوقت بقا هنتكلم علي ال k-nearest neighbors ..

Def: classification

DataSet $D = \{(x, y)\}$

For all $x \in D$ -> the x vectors are real valued vectors of length m (features (attributes)/input)

For all $y \in D$ -> the y comes from a subset $\{1, 2, 3, \dots, L\}$ (labels (classes)/output)

$M = \# \text{ of features}$

$N = \# \text{ of examples}$

بكدا نقدر نعرف ال binary classifier كإئو extension of classification .. تعال نعرف كمان hypothesis .. ك decision rule .. في التريننج
 احنا هدفنا نتعلم hypothesis h .. وفي التست تايم احنا هدفنا نجيب ال \hat{y}

k-NN + Model Selection

Wednesday, January 29, 2020 9:56 AM

Def: Classification

$$D = \{(\vec{x}^{(i)}, y^{(i)})\}_{i=1}^N$$

$\forall i \vec{x}^{(i)} \in \mathbb{R}^M$ ← Feature (attributes)/input
 ← real valued vectors of length M

$\forall i y^{(i)} \in \{1, 2, \dots, L\}$ ← label (classes)/output
 ← y

$M = \# \text{ of features}$
 $N = \# \text{ of examples} = |D|$

Def: Binary Classification

Above where $y^{(i)} \in \{0, 1\}$
 $\in \{+1, -1\}$
 $\in \{T, F\}$
 $\in \{\text{cat}, \text{dog}\}$

$|y| = 2$

Def: Hypothesis (aka. Decision Rule)
 for Binary Class.

$$h: \mathbb{R}^M \rightarrow \{+, -\}$$

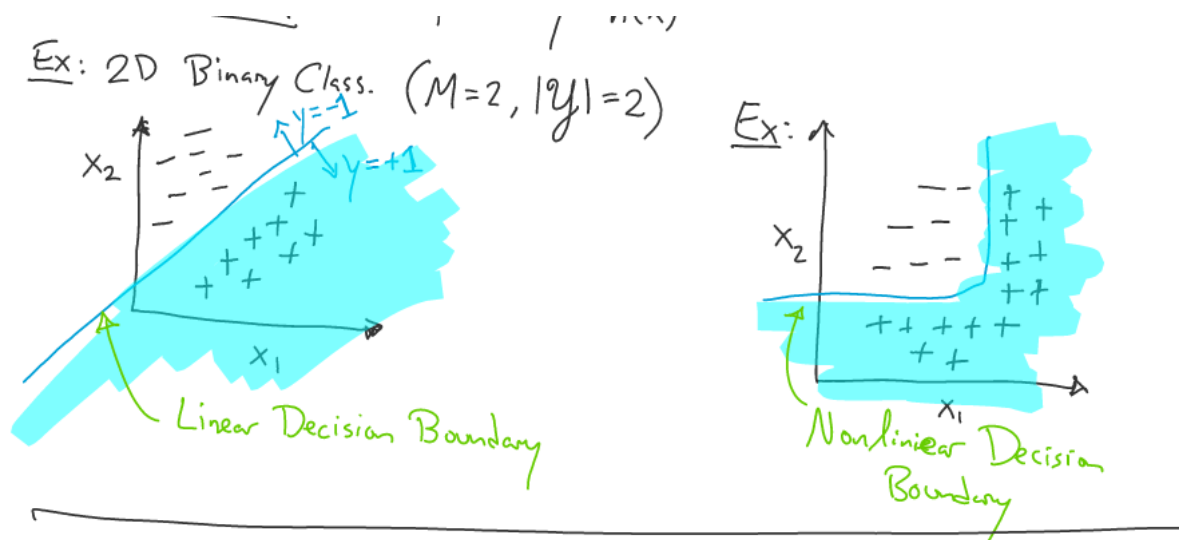
Train time learn h

Test time Given \vec{x} , predict $\hat{y} = h(\vec{x})$

تعال نفكر في ال hypothesis .. pictorially ... انا مفهمتش الدكتور قصدو ايه بالكلمه ديه بس لما ترجمتها لاقيتها بالصور .. تقريباً قصدو نتخيل ال hypothesis او اننا نتصورها يعني .. هيبان مع الوقت ..

Ex: 2D binary classification ($M = 2, |y| = 2$) .. طالما باينري كلاسيفيكيشن .. بيقتا حجم .. طالما هي حاجه في ال 2 دي .. كذا انت عندك 2 فيتشرز .. طالما باينري كلاسيفيكيشن .. بيقتا حجم .. الواي بتاعك هيبقا 2 ..

احنا دلوقت نقدر نرسم صوره ...



تعال نعرف A nearest neighbor classifier .. هيقا ليه predict method و train method .. الترين هنا هيخزن الدات سبت D .. ال
 prediction هت assign the label of the nearest point in our dataset D .. تعال نشوف ده هيجصل ازاي لو عندنا داتا سبت في ال 2D ..
 هنقيس الداتا ازاي ..

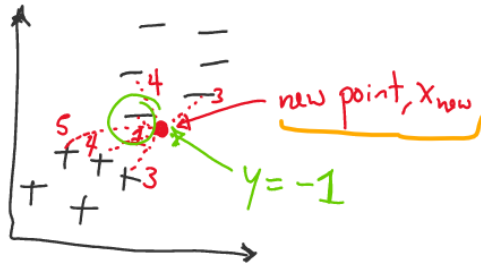
K-Nearest Neighbors Classifier

def train(D): store D

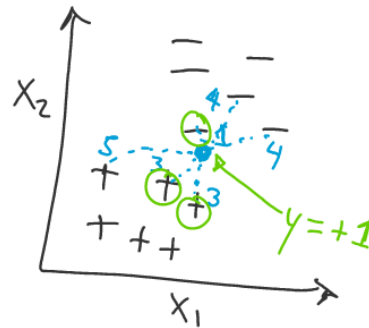
def predict(x):

Assign the most common
 label of the nearest points in D
 \uparrow
 k

Ex: NN



Ex: KNN k=3



لو ال K = 3 .. هتلاقي عندك أقرب 3 نقط هم ال 2 و ال 1 - .. فكذا ال y = 1 ..

KNN: Remarks

Distance Functions:

- KNN requires a **distance function**

$$g : \mathbb{R}^M \times \mathbb{R}^M \rightarrow \mathbb{R}$$

- The most common choice is **Euclidean distance**

$$g(\mathbf{u}, \mathbf{v}) = \sqrt{\sum_{m=1}^M (u_m - v_m)^2}$$

- But other choices are just fine (e.g. **Manhattan distance**)

$$g(\mathbf{u}, \mathbf{v}) = \sum_{m=1}^M |u_m - v_m|$$

دلوقت احنا محتاجين نفكر شويه في اللي الألو رزم بيعملو .. اللي احنا بنعملو علي ال board هو ال Euclidean distance .. في اختيارات تانيه عادي .. اللي هو Manhattan distance ... لو بتروح من نقطه u لنقطه v ... متقدرش تمشي علي طول .. في شبكه عندك .. فهتمشي قدام شويه وبعدين تلف يمين وبعدين تلف تاني وبعدين تمشي علي طول هتلاقيك رحت علي طول .. فعشان كذا بتاخذ ال sum ال absolute values بتاعت ال dimensions اللي احنا شغالين عليها ...

هنرجع للسؤال بتاع ازاي ن handle التعادل .. يعني مثلا لو كان في المثال الي فات $k = 6$ بدل $k = 3$.. كان هيبقا في تعادل 3 - و 3 + ... ففي حد اقترح انك ترجع بصهرك لورا خطوه بدل $k = 6$ خذ $k = 5$.. او اطلع بوشك لقدام .. خذ $k = 7$.. في حد اقترح حل فشيخ .. استخدم ال مسافات ك weights و اعمل weghited majority votes ... في حل تاني .. خذ distance مختلف ...

تاني سؤال عندنا ... ايه هو ال inductive bias بتاع ال K Nearest neighbor .. ال inductive bias هو ال principle by which we generalize to unseen data .. و هنا نقدر نفكر ايه هو ال inductive bias لل KNN ..

ال principle اللي عن طريقه بيحصل generalization هنا هو ال assumption ان الداتا مش كلها عباره عن + او - ... انما هم بينفصلو عن بعض .. وعشان هم منفصلين عن بعض .. فنقدر نطلع ب principle ان بيحصل generalization .. ازاي نعمل predictions based on the fact that the data aren't all just mixed up .. ايه هو ال principle تحت حاجه زي كذا .. في term اسمو spatial autocorrelation ده ترم في ال geography .. بيقول ان الحاجات اللي قريبه من بعض بتبقا قريبه من بعض .. similar points بييفا ليهم similar labels ... او بمعنى اصح

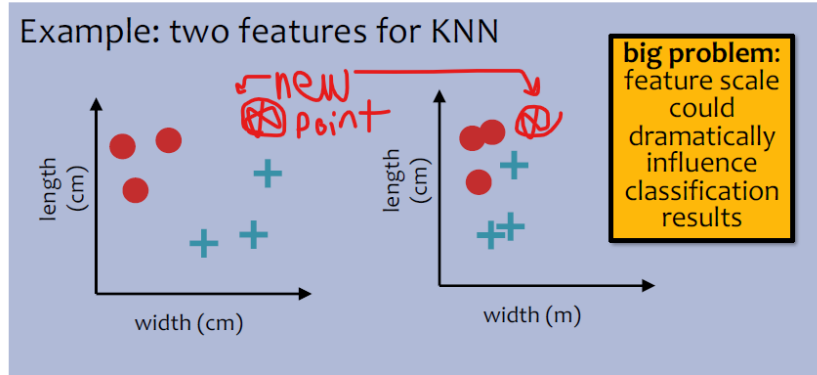
Nearby points should have similar labels

في inductive bias هنا برضو ويعتبر أحسن سيكه .. وده ان KNN بتفترض ان كل ال dimensions are created equally .. افترض ان عندك 2 فيتشرز .. زي مثلا الطول و الويدث لحاجه معينه .. و هم متقاسين بال CM .. وحد قام معدل علي الداتا سبت بتاعتك و خلي الاطوال بال m ... فتخيل ان في نقطه جديده جاتك ...

KNN: Remarks

Inductive Bias:

1. Similar points should have similar labels
2. All dimensions are created equally!



34

فتخيل ان النقطه لو كان ال scale لسه بال cm كانت هتبقا أكيد أكيد +ve .. دلوقت كذا ممكن يحصل اختلاط وتبقا -ve .. فبالتالي المقصود هنا ان ال feature scale بيأثر بشكل كبير في ال classification result .. لل KNN classifier

KNN: Remarks

Computational Efficiency:

- Suppose we have N training examples, and each one has M features
- Computational complexity for the special case where $k=1$:

Task	Naive	k-d Tree
Train	$O(1)$	$\sim O(M N \log N)$
Predict (one test example)	$O(MN)$	$\sim O(2^M \log N)$ on average

Problem: Very fast for small M , but very slow for large M

In practice: use stochastic approximations (very fast, and empirically often as good)

35

هنعوز نفكر شويه في ال computational efficiency .. أول implementation اللي هو ال Naive .. عندك N examples with M features .. ده اللي هو انك تخزن الداتا .. أوردت أوف 1 .. انما ال cost بكام انك ت predict مثال جديد .. 2 فور لوب ... اللي برا هتلف علي الأمثلة اللي عندك .. الي جوا هتلف علي كل فيتشر جوا كل مثال عندك ... عشان ال distance function هتلف علي كل ال features اللي عندك .. فبالتالي ال prediction هيبقا ليه أوردت $O(MN)$.. ايه اللي هيجصل لو عندك 100 مليون مثال .. كل ما تيجي تعمل prediction .. انت هتلف عليهم كلهم .. في حل ثاني ...

في نقط بعيد اصلاً جداً عن الي انت بتتعامل معاه دلوقت .. انت مش محتاج قياسهم ف حاجه .. فحد عمل كذا و قال انا هبني حاجه اسمها KD tree .. هتبني data structure .. و تتكون من hypercube high dimensional space .. وديه بتقسم ال space لشوية مكعبات .. فانت بتعمل شغل في الأول وانت بتبني ال KD tree ... فانت خدت $O(MN \log(N))$... بس دلوقت انت بقا عندك tree of hypercube .. بس دلوقت انت عندك نقط أقل من اللي كان عندك في ال naive لإنهم بعدا عنك جداً .. فبالتالي ال prediction هيبقا $O(2^M * \log(N))$ فانت بقيت سريع .. بس في مشكله هنا ... وهو ال 2^M ... لو انت عندك انفورميشن كتيره فشاخه لكل مريض .. انت هتبقا بطيء جداً جداً جداً جداً .. فهي بتشتغل كويس بس لما يكون عدد ال features الي حد ما معقول .. في ارض الواقع انت لو هت implement ال KNN متستخدمش ولا حاجه من اللي اتقالت .. اومال اعمل ايه ..

زي بقيت الناس .. استخدم stochastic approximation .. بشكل سريع خمن ال K nearest neighbors من اللي حواليك .. هي مش هتبقا حاجه فشخه بس هتبقا قريبه من اللي بيحصل لو استخدمت الطرق اللي شرحناها فوق ..

.....

في الكورس بتاعنا هنتكلم شوية علي ال theoretical guarantees ل algorithms اللي بنتكلم عنهم ...

KNN: Remarks

Theoretical Guarantees:

Cover & Hart (1967)

Let $h(x)$ be a Nearest Neighbor ($k=1$) binary classifier. As the number of training examples N goes to infinity...

$$\text{error}_{\text{true}}(h) < 2 \times \text{Bayes Error Rate}$$

"In this sense, it may be said that half the classification information in an infinite sample set is contained in the nearest neighbor."

very informally, Bayes Error Rate can be thought of as: 'the best you could possibly do'

36

احنا هنتكلم علي حاجه اسمها PAC learning اللي هو هتحدد شوية guarantees اللي هت apply to all of the different binary classification الجورنمز اللي اتكلمنا عنهم .. و كمان هن follow up .. شوية النظريات اللي عندنا مع شوية proofs .. احنا مش هنخش في البروفس انما هنتكلم من فوق كذا علي النظرية من غير أي اثبات ... الدكتور بيتكلم ع الصورة اللي فوق .. ديه بيبر من 1967 .. افترض ان ال $h(x)$ ديه هو ال nearest neighbor classifier .. ف ده ال $k=1$. وده باينري كلاسيفير .. فبيقولو كل ما عدد الأمثلة زادت و راح للمالا نهاية .. ال true error .. اللي هو a quantity that we can't actually measure .. هو ليه boundary .. وهو ال Bayes error rate * 2 .. ال bayes error rate .. ده اللي هو أحسن حاجه تقدر تعملها لأي كلاسيفير علي الداتا سيت اللي عندك .. ده معناه ايه .. انت عندك داتا سيت مثلاً و احسن حاجه تقدر تعملها هي ال 5% إيرور .. ال theoretical result بتقول إن ال true error .. هيبقا اقل من 10% إيرور ... فده فعلاً حاجه مهمه ..

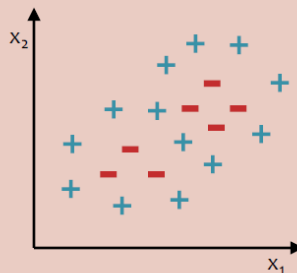
Decision Boundary Example

Dataset: Outputs $\{+, -\}$; Features x_1 and x_2

In-Class Exercise

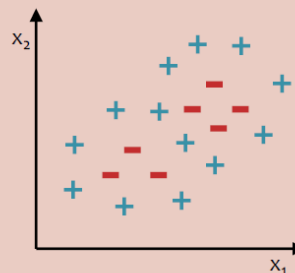
Question 1:

- A. Can a **k-Nearest Neighbor classifier with $k=1$** achieve **zero training error** on this dataset?
- B. If 'Yes', draw the learned decision boundary. If 'No', why not?



Question 2:

- A. Can a **Decision Tree classifier** achieve **zero training error** on this dataset?
- B. If 'Yes', draw the learned decision bound. If 'No', why not?



38

دلوقت احنا عندنا ال decision boundary .. ازاي نقدر نوصف ال KNN أو ال decision tree كصوره بتوصف الي احنا بنعملو .. ازاي ال algorithm بي behave في حاجه 2D ... تعال نبص علي سؤال رقم 2 في الصورة الي فوق ديه .. اول اجابه اللي هي A هتبقا yes .. ليه ... ممكن نرسم دايرتين حوالين السالب اللي فوق ... ازاي هنعمل كذا اصلاً .. تخيل انت ف 1982 .. و تعال ن pixelate الصورة اللي عندنا ... عندنا شوية pixels .. هنسأل سؤال لكل pixle .. لو احنا عندنا نقطه جديده بتظهر علي pixel location .. لو كان لونها احمر خليها احمر لو ازرق خليها ابيض .. هنمشي

column column .. فهتقوم ملون ال pixel كلها باللون الاحمر .. فتهفضل ماشي عادي لحد ما يبقا عندك دايرتين .. و أي حاجه بقا لونها احمر و أي نقطه هتيجي عليها .. هبيقا class = red ... وده فعلاً هي crrect classifiy ..

بالنسبة لسؤال رقم 3 ... الاجابه yes .. ليه ؟ .. مغيث وقت جابو انت . إسأل نفسك أنهي نوع من انواع الشجر نقدر نبنيه ..

في تفصيله كمان ... لو انت فعلا عندك البكسلز بالمنظر ده .. مش هتبقا دواير سمووث .. انما اسمها Voronoi diagram ده اللي بيطلعك اللاينز اللي فعلا بتفصل ما بين المناطق

Decision Boundary Example

Dataset: Outputs {+, -}; Features x_1 and x_2

In-Class Exercise

Question 2

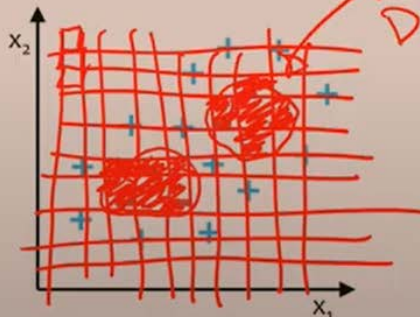
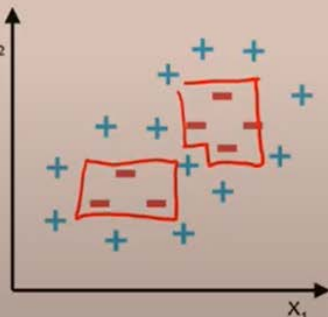
A. Can a **k-Nearest Neighbor classifier with $k=1$** achieve **zero training error** on this dataset?

B. If 'Yes', draw the learned decision boundary. If 'No', why not?

Question 3

A. Can a **Decision Tree classifier** achieve **zero training error** on this dataset?

B. If 'Yes', draw the learned decision boundary for the Decision Tree classifier. If 'No', why not?

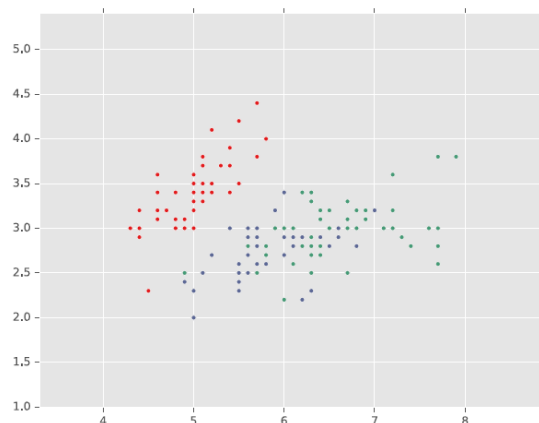



A = Yes
B = No
C = cakmity

عشان صعب اننا نرسم الحاجات ديه بايدينا .. تعال نبص علي شوية جرافات بالكمبيوتر ..

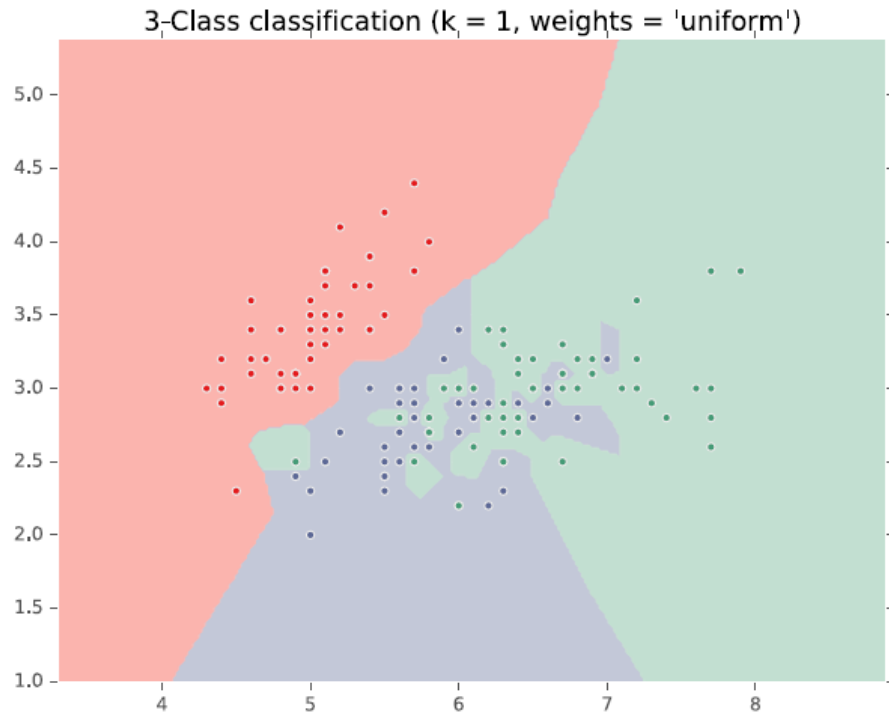
ديه الداتا اهي

KNN on Fisher Iris Data



KNN on Fisher Iris Data

Special Case: Nearest Neighbor



46

شوف السلايدز عشان في صور كتيره ورا بعضها .. دلوقت عاوزن بشكل بسيط كدا نلخص ال model selection .. ازاي نختار a particular scision tree .. و نقول أنهي واحده هي الاحسن ..