CLASSIFICATION

Introducción a la Ciencia de Datos

Some of the figures in this presentation are taken from: An Introduction to Statistical Learning, with applications in R" (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani.

Some slides are based on Abbass Al Sharif's slides for his course DSO 530: Applied Modern Statistical Learning Techniques.

Overview of Classification

- Examples:
  1. An online banking service must be able to determine whether or not a transaction being performed on the site is fraudulent, on the basis of the user's IP address, past transaction history, and so on.
  2. A person arrives at the emergency room with a set of symptoms that could possibly be attributed to one of three medical conditions. Which of the three conditions does the individual have?
  3. On the basis of DNA sequence data for a number of patients with and without a given disease, a biologist would like to figure out which DNA mutations are deleterious (disease-causing) and which are not.

Overview of Classification

- We will always assume we have observed n data points x_i, where $i = 1, 2, \ldots, n$. These observations –i.e., instances or examples– are called *training data* because we will use these observations to "teach" –i.e., train– our model to estimate f.

- Let x_{ij} represent the value of the j^{th} predictor, or input, for observation i, where $j = 1, 2, \ldots, p$.

- Correspondingly, let y_i represent the response variable for the i^{th} observation –i.e., prediction or target. Then our training data consist of $\{(x_1, y_1),(x_2, y_2), \ldots,(x_n, y_n)\}$ where $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})^T$.

Overview of Classification

- We aim to apply a *classification method* to the training data to estimate the unknown function *f*.

- In other words, we want to find a function f^* such that $Y \approx f^*(X)$ for any observation (X,Y) –i.e., a classification model.

- In general, we do not really care how well the classification method works training on the training data. Instead, we are interested in the accuracy of the predictions that we obtain when we apply our method to previously unseen *test data*.

Types of predictor variables

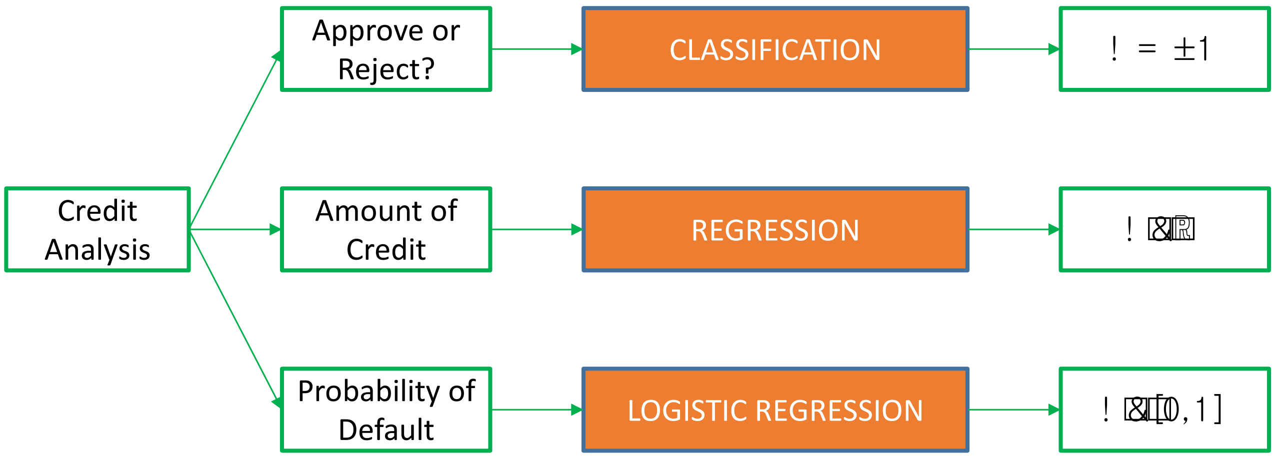
- Qualitative:
  - Nominal
  - Ordinal
- Quantitative:
  - Discrete
  - Continuous

The **target** in classification is qualitative, usually nominal and uni-dimensional.

Overview of Classification

Three learning problems

- What are the simplest models to solve these learning tasks?



X. Bresson (2023). Lecture 4 : Linear Models and Support Vector Machine. Course CS3244 – Machine Learning.

More examples…

Iris

- *Objective*: To predict class for (unknown) flowers from shape measurements
  - class = {iris setosa, iris virginica, iris versicolor}
- *Predictors* (x4):
  - Petal length
  - Petal width
  - Sepal length
  - Petal length

More examples…

Iris

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5,1	3,5	1,4	0,2	setosa
4,9	3,0	1,4	0,2	setosa
4,7	3,2	1,3	0,2	setosa
4,6	3,1	1,5	0,2	setosa
5,0	3,6	1,4	0,2	setosa
5,4	3,9	1,7	0,4	setosa
4,6	3,4	1,4	0,3	setosa
5,0	3,4	1,5	0,2	setosa
4,4	2,9	1,4	0,2	setosa
4,9	3,1	1,5	0,1	setosa
5,4	3,7	1,5	0,2	setosa
4,8	3,4	1,6	0,2	setosa
4,8	3,0	1,4	0,1	setosa
4,3	3,0	1,1	0,1	setosa
5,8	4,0	1,2	0,2	setosa
5,7	4,4	1,5	0,4	setosa
5,4	3,9	1,3	0,4	setosa
5,1	3,5	1,4	0,3	setosa
5,7	3,8	1,7	0,3	setosa
5,1	3,8	1,5	0,3	setosa
5,4	3,4	1,7	0,2	setosa
5,1	3,7	1,5	0,4	setosa
4,6	3,6	1,0	0,2	setosa
5,1	3,3	1,7	0,5	setosa

```
summary(iris_data)
```

```
  Sepal.Length    Sepal.Width     Petal.Length    Petal.Width   
 Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100  
 1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300  
 Median :5.800   Median :3.000   Median :4.350   Median :1.300  
 Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199  
 3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800  
 Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500  
       Species  
 setosa    :50  
 versicolor:50  
 virginica :50  
```

iris.qmd

More examples…

Breast Cancer

- *Objective*: To predict whether an individual has a benign or malignant tumour based on their characteristics
  - diagnosis = {‘B’, ‘M’}
- *Predictors* (x30)
  - https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data

More examples…

Breast Cancer

cancer.qmd

id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	points_mean	symmetry_mean	dimension_mean	radius_se	texture_se	perimeter_se
87139402	B	12.32	12.39	78.85	464.1	0.1028	0.06981	0.03987	37	0.1959	0.05955	236	0.6656	1.67
8910251	B	10.6	18.95	69.28	346.4	0.09688	0.1147	0.06387	0.02642	0.1922	0.06491	0.4505	1.197	3.43
905520	B	11.04	16.83	70.92	373.2	0.1077	0.07804	0.03046	0.0248	0.1714	0.0634	0.1967	1.387	1.342
868871	B	11.28	13.39	73	384.8	0.1164	0.1136	0.04635	0.04796	0.1771	0.06072	0.3384	1.343	1.851
9012568	B	15.19	13.21	97.65	711.8	0.07963	0.06934	0.03393	0.02657	0.1721	0.05544	0.1783	0.4125	1.338
906539	B	11.57	19.04	74.2	409.7	0.08546	0.07722	0.05485	0.01428	0.2031	0.06267	0.2864	1.44	2.206
925291	B	11.51	23.93	74.52	403.5	0.09261	0.1021	0.1112	0.04105	0.1388	0.0657	0.2388	2.904	1.936
87880	M	13.81	23.75	91.56	597.8	0.1323	0.1768	0.1558	0.09176	0.2251	0.07421	0.5648	1.93	3.909
862989	B	10.49	19.29	67.41	336.1	0.09989	0.08578	0.02995	0.01201	0.2217	0.06481	355	1.534	2.302
89827	B	11.06	14.96	71.49	373.9	0.1033	0.09097	0.05397	0.03341	0.1776	0.06907	0.1601	0.8225	1.355
91485	M	20.59	21.24	137.8	1320	0.1085	0.1644	0.2188	0.1121	0.1848	0.06222	0.5904	1.216	4.206
8711003	B	12.25	17.94	78.27	460.3	0.08654	0.06679	0.03885	0.02331	197	0.06228	0.22	0.9823	1.484
9113455	B	13.14	20.74	85.98	536.9	0.08675	0.1089	0.1085	0.0351	0.1562	0.0602	0.3152	0.7884	2.312
857810	B	13.05	19.31	82.61	527.2	0.0806	0.03789	0.000692	0.004167	0.1819	0.05501	404	1.214	2.595
9111805	M	19.59	25	127.7	1191	0.1032	0.09871	0.1655	0.09063	0.1663	0.05391	0.4674	1.375	2.916
925277	B	14.59	22.68	96.39	657.1	0.08473	133	0.1029	0.03736	0.1454	0.06147	0.2254	1.108	2.224
867387	B	15.71	13.93	102	761.7	0.09462	0.09462	0.07135	0.05933	0.1816	0.05723	0.3117	0.8155	1.972
89511502	B	12.67	17.3	81.25	489.9	0.1028	0.07664	0.03193	0.02107	0.1707	0.05984	0.21	0.9505	1.566
89263202	M	20.09	23.86	134.7	1247	108	0.1838	0.2283	128	0.2249	0.07469	1.072	1.743	7.804
866714	B	12.19	13.29	79.08	455.8	0.1066	0.09509	0.02855	0.02882	188	0.06471	0.2005	0.8163	1.973
874373	B	11.71	17.19	74.68	420.3	0.09774	0.06141	0.03809	0.03239	0.1516	0.06095	0.2451	0.7655	1.742
919812	B	11.69	24.44	76.37	406.4	0.1236	0.1552	0.04515	0.04531	0.2131	0.07405	0.2957	1.978	2.158
904971	B	10.94	18.59	70.39	370	0.1004	0.0746	0.04944	0.02932	0.1486	0.06615	0.3796	1.743	3.018
866458	B	15.1	16.39	99.58	674.5	115	0.1807	0.1138	0.08534	0.2001	0.06467	0.4309	1.068	2.796
864292	B	10.51	20.19	68.64	334.2	0.1122	0.1303	0.06476	0.03068	0.1922	0.07782	0.3336	1.86	2.041
859983	M	13.8	15.79	90.43	584.1	0.1007	128	0.07789	0.05069	0.1662	0.06566	0.2787	0.6205	1.957
862009	B	13.45	18.3	86.6	555.1	0.1022	0.08165	0.03974	0.0278	0.1638	0.0571	295	1.373	2.099
852973	M	15.3	25.27	102.4	732.4	0.1082	0.1697	0.1683	0.08751	0.1926	0.0654	439	1.012	3.498
898143	B	9.606	16.84	61.64	280.5	0.08481	0.09228	0.08422	0.02292	0.2036	0.07125	0.1844	0.9429	1.429
9010877	B	13.4	16.95	85.48	552.4	0.07937	0.05696	0.02181	0.01473	165	0.05701	0.1584	0.6124	1.036
893548	B	13.05	13.84	82.71	530.6	0.08352	0.03735	0.004559	0.008829	0.1453	0.05518	0.3975	0.8285	2.567

More examples…

Titanic

- *Objective*: To predict whether (unknown) passengers survive based on their characteristics
  - survived = {0, 1}
- *Predictors* (x9): age, gender, travelling class, etc.
  - pclass = {1st, 2nd, 3rd}
  - sex = {male, female}
  - age = {0, 1, …, 100}
  - sibsp: # siblings on board (brother/sister, husband/wife)
  - parch: # direct siblings on board (father/mother, son/daughter)
  - ticket number
  - care = [0, 100]
  - cabin number
  - embarked: departing harbour

More examples…

Titanic

titanic.qmd

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7.925		S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1	C123	S
5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
6	0	3	Moran, Mr. James	male		0	0	330877	8.4583		Q
7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21.075		S
9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333		S
10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14	1	0	237736	30.0708		C
11	1	3	Sandstrom, Miss. Marguerite Rut	female	4	1	1	PP 9549	16.7	G6	S
12	1	1	Bonnell, Miss. Elizabeth	female	58	0	0	113783	26.55	C103	S
13	0	3	Saundercock, Mr. William Henry	male	20	0	0	A/5. 2151	8.05		S
14	0	3	Andersson, Mr. Anders Johan	male	39	1	5	347082	31.275		S
15	0	3	Vestrom, Miss. Hulda Amanda Adolfina	female	14	0	0	350406	7.8542		S
16	1	2	Hewlett, Mrs. (Mary D Kingcome)	female	55	0	0	248706	16		S
17	0	3	Rice, Master. Eugene	male	2	4	1	382652	29.125		Q
18	1	2	Williams, Mr. Charles Eugene	male		0	0	244373	13		S
19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Vandemoortele)	female	31	1	0	345763	18		S
20	1	3	Masselmani, Mrs. Fatima	female		0	0	2649	7.225		C

K-NN

Classification Methods

Understanding classification using NN

- *Nearest Neighbor (NN) Classifiers* are defined by their characteristic of classifying unlabeled examples by assigning them the class of the most similar labeled examples.
- NN classifiers are well-suited for classification tasks where relationships among the features and the target classes are difficult to understand, yet the items of similar class type tend to be fairly homogeneous.
- If there is not a clear distinction among the groups, the algorithm is by and large not well-suited for identifying the boundary.

The k-NN algorithm

- The k-NN algorithm begins with a training dataset containing examples that are classified into several categories, as labeled by a nominal variable.
- Assume that we have a *test dataset* containing unlabeled examples that otherwise have the same features as the *training data*.
- For each record in the test dataset, k-NN identifies *k* records in the training data that are the "nearest" in distance/similarity, where *k* is an integer specified in advance.
- The unlabeled test instance is assigned the class of the majority of the *k* nearest neighbors.

The k-NN algorithm

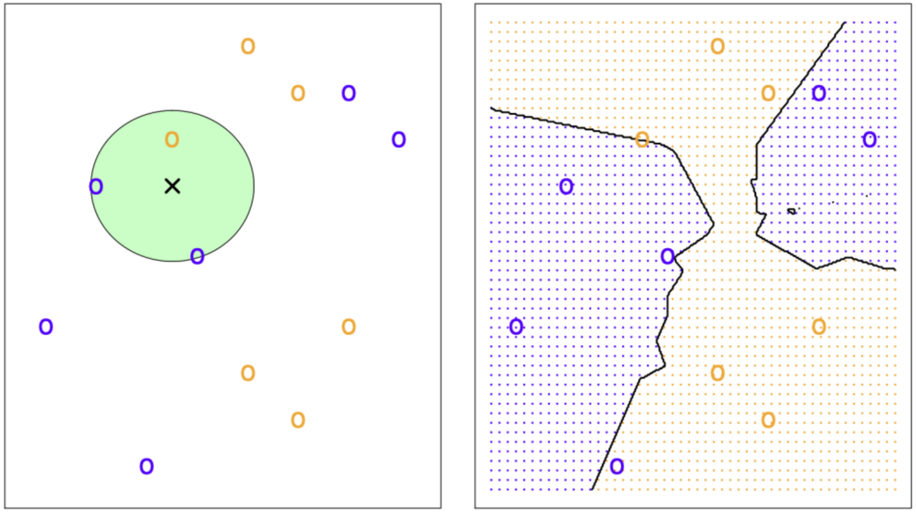


FIGURE 2.14. *The KNN approach, using $K = 3$, is illustrated in a simple situation with six blue observations and six orange observations.* Left: *a test observation at which a predicted class label is desired is shown as a black cross. The three closest points to the test observation are identified, and it is predicted that the test observation belongs to the most commonly-occurring class, in this case blue.* Right: *The KNN decision boundary for this example is shown in black. The blue grid indicates the region in which a test observation will be assigned to the blue class, and the orange grid indicates the region in which it will be assigned to the orange class.*

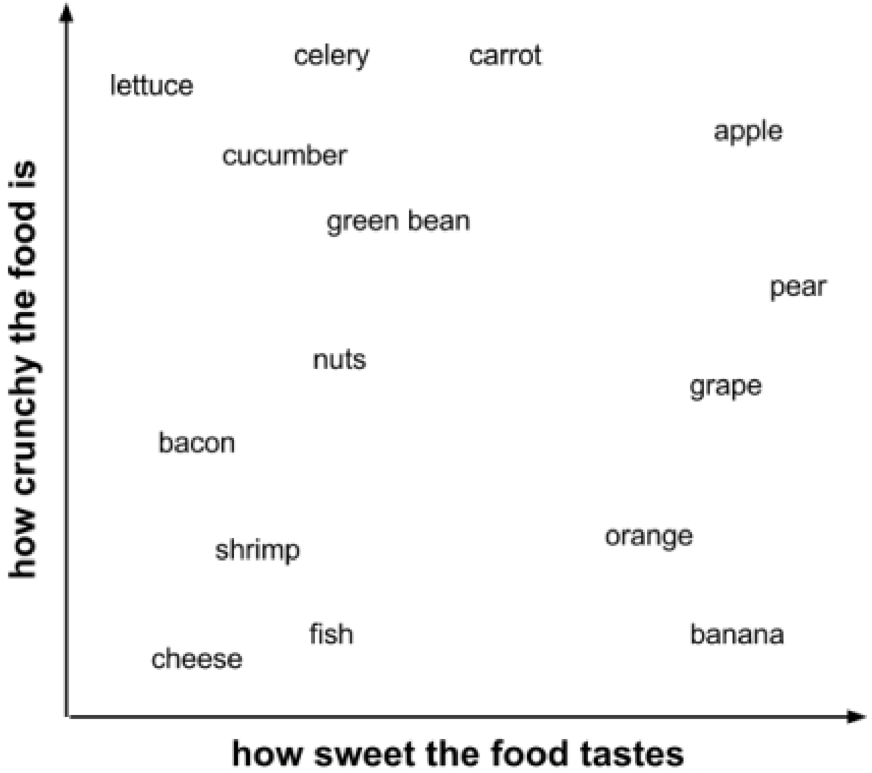
The k-NN algorithm

- *Dataset*: Blind tasting experience:
  - Only two features of each ingredient is recorded: (1) a measure from 1 to 10 of how crunchy the ingredient is, and (2) a measure from 1 to 10 score of how sweet the ingredient tastes.
  - We then labeled each ingredient as one of three types of food: fruits, vegetables, or proteins.

ingredient	sweetness	crunchiness	food type
apple	10	9	**fruit**
bacon	1	4	**protein**
banana	10	1	**fruit**
carrot	7	10	**vegetable**
celery	3	10	**vegetable**
cheese	1	1	**protein**

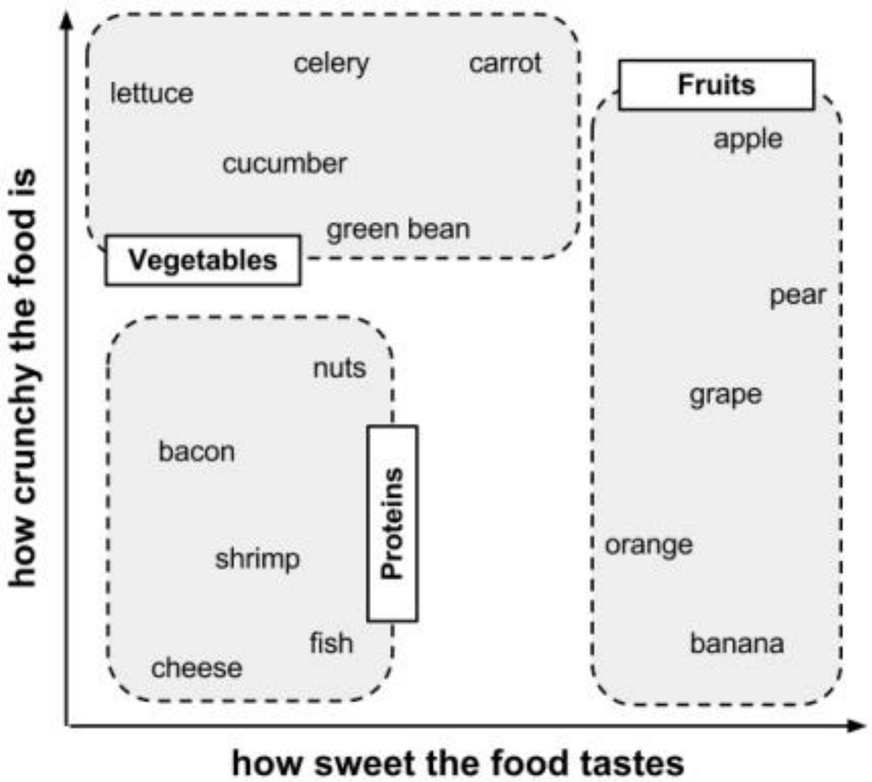
The k-NN algorithm

- The k-NN algorithm treats the features as coordinates in a multidimensional *feature space*:



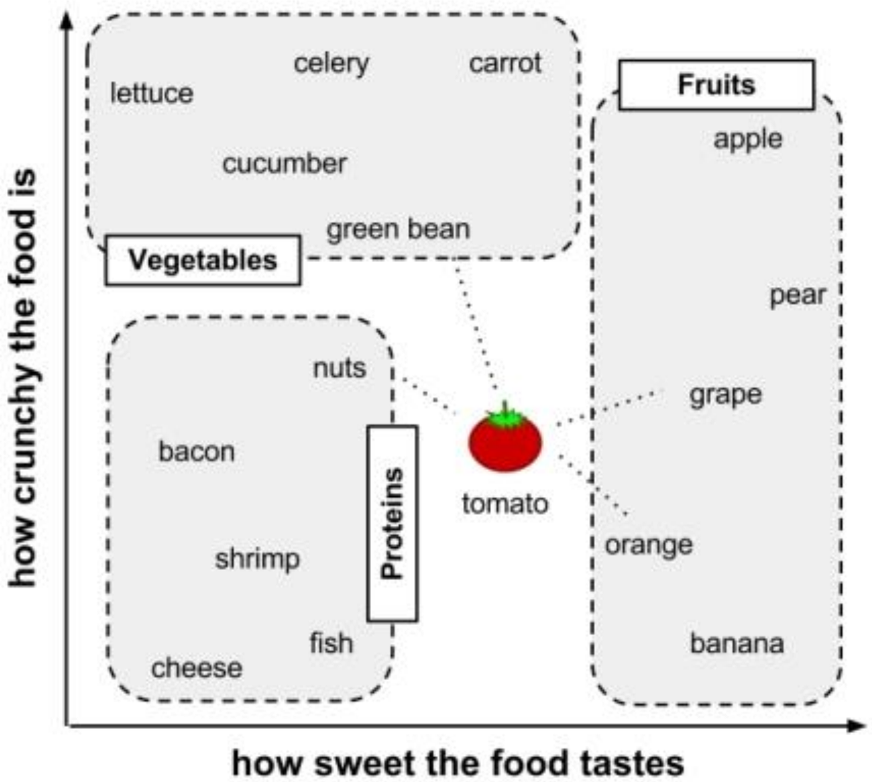
The k-NN algorithm

- Similar types of food tend to be grouped closely together:



The k-NN algorithm

- Use k-NN to settle the age-old question: is a tomato a fruit or a vegetable?



The k-NN algorithm

- Locating the tomato's nearest neighbors requires a *distance function*, or a formula that measures the similarity between two instances.
- Traditionally, the k-NN algorithm uses *Euclidean distance*:

$$dist(X_1, X_2) = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + \cdots + (x_{1n} - x_{2n})^2}$$

- where X_1 and X_2 are the examples to be compared, each having *n* features. The term x_{11} refers to the value of the first feature of example X_1, while x_{21} refers to the value of the first feature of example X_2.

The k-NN algorithm

- For example, to calculate the distance between the tomato (sweetness = 6, crunchiness = 4), and the green bean (sweetness = 3, crunchiness = 7), we can use the formula as follows:

$$dist(tomato, greenbean) = \sqrt{(6-3)^2 + (4-7)^2} = 4.2$$

ingredient	sweetness	crunchiness	food type	distance to *tomato*
grape	8	5	fruit	2.2
green bean	3	7	vegetable	4.2
nuts	3	6	protein	3.6
orange	7	3	fruit	1.4

The k-NN algorithm

- The predicted class can be the one with ”the most votes” according to k = 4 neighbours, e.g.:

$$\Pr(Y = j | X = x_0) = \frac{1}{K} \sum_{i \in \mathcal{N}_0} I(y_i = j).$$

ingredient	sweetness	crunchiness	food type	distance o-f *tomato*
grape	8	5	fruit	2.2
green bean	3	7	vegetable	4.2
nuts	3	6	protein	3.6
orange	7	3	fruit	1.4

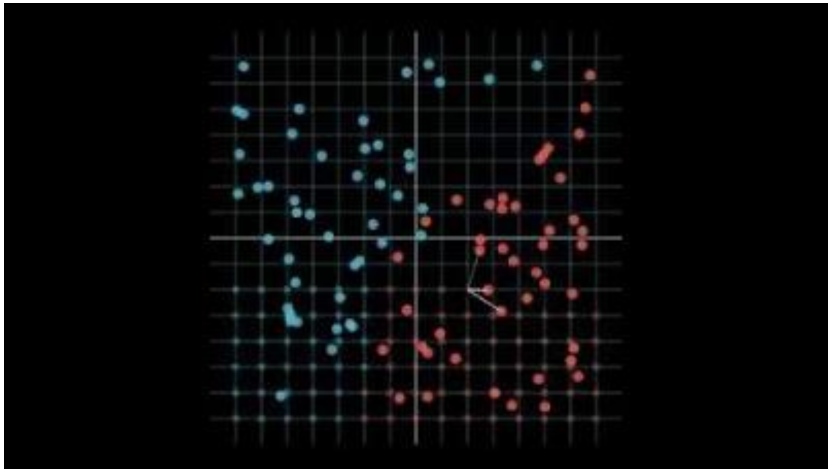
$$\Pr(Y = fruit | X = tomato) = \frac{2}{4}$$

$$\Pr(Y = vegetable | X = tomato) = \frac{1}{4}$$

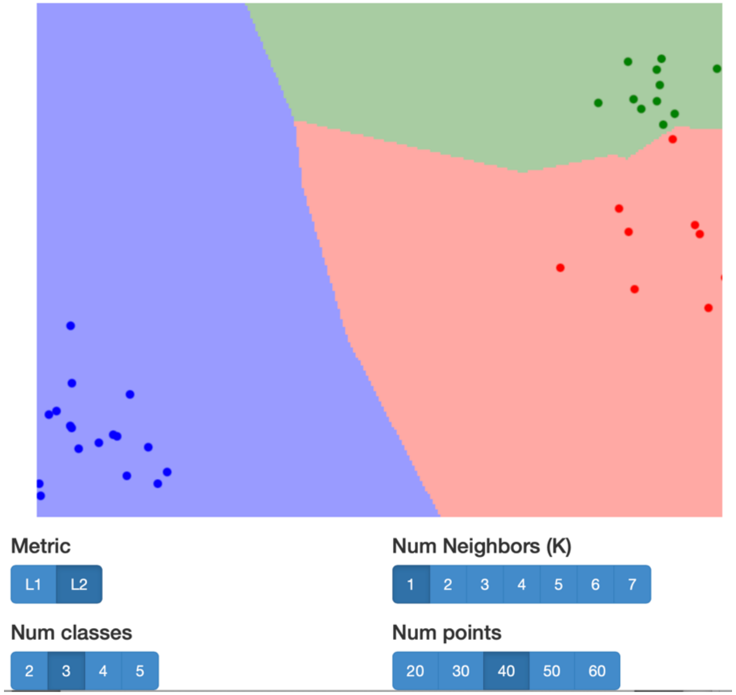
The k-NN algorithm

- Decision boudaries:

https://youtu.be/Mhv-HxGSgHU?si=8jc3pBtU7L5Z4yW0&t=140



http://vision.stanford.edu/teaching/cs231n-demos/knn/



The k-NN algorithm

- Training and validation
  - No specific procedure for training or validation
  - Still, we can calculate classification error for training data
    - Set a K value (>1)
    - Check if each training instance would be correctly classified if considering its neighbours (optionally, itself)
    - Calculate metrics, e.g., accuracy
  - Same for validation!

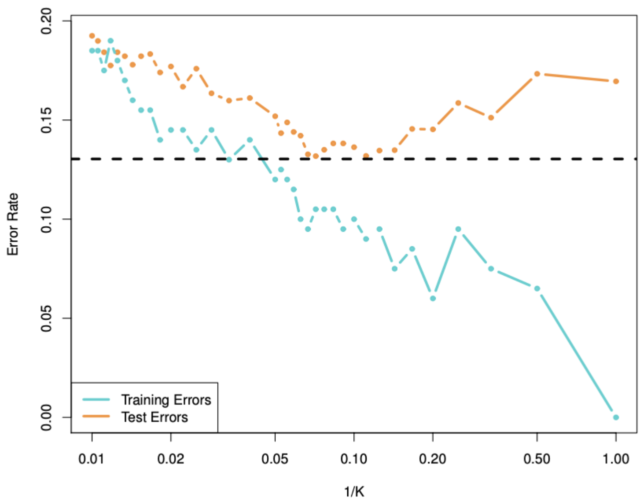


FIGURE 2.17. *The KNN training error rate (blue, 200 observations) and test error rate (orange, 5,000 observations) on the data from Figure 2.13, as the level of flexibility (assessed using $1/K$ on the log scale) increases, or equivalently as the number of neighbors K decreases. The black dashed line indicates the Bayes error rate. The jumpiness of the curves is due to the small size of the training data set.*

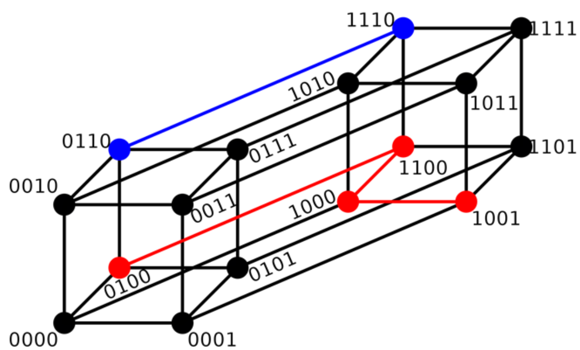
Impact of distance measures

Name	Definition
Average (L_1, L_∞)	$\frac{\sum_{i=1}^{n} \lvert x_i-y_i\rvert+\max_i \lvert x_i-y_i\rvert}{2}$
Kumar-Johnson	$\sum_{i=1}^{n}\left(\frac{(x_i^2+y_i^2)^2}{2(x_i y_i)^{3/2}}\right)$
Taneja	$\sum_{i=1}^{n}\left(\frac{x_i+y_i}{2}\right) ln \left(\frac{x_i+y_i}{2\sqrt{x_i y_i}}\right)$
Pearson	$1-\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2 \star \sum_{i=1}^{n}(y_i-\bar{y})^2}}$ $\bar{x}=\frac{1}{n}\sum_{i=1}^{n} x_i$
Correlation	$\frac{1}{2}\left(1-\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}\right)$
Squared Pearson	$1-\left(\frac{\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i-\bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i-\bar{y})^2}}\right)^2$
Hamming	$\sum_{i=1}^{n} 1_{x_i \neq y_i}$
Hausdorff	$\max(h(x,y),h(y,x))$ $h(x,y)=\max_{x_i\in x}\min_{y_i \in y}\lvert\lvert x_i-y_i\rvert\rvert$
χ^2 statistic	$\sum_{i=1}^{n}\frac{x_i-m_i}{m_i},\ m_i=\frac{x_i+y_i}{2}$
Whittaker's index of assoc.	$\frac{1}{2}\sum_{i=1}^{n}\lvert\frac{x_i}{\sum_{i=1}^{n} x_i}-\frac{y_i}{\sum_{i=1}^{n} y_i}\rvert$
Meehl	$\sum_{i=1}^{n-1}(x_i-y_i-x_{i+1}+y_{i+1})^2$
Motyka	$\frac{\sum_{i=1}^{n}\max(x_i,y_i)}{\sum_{i=1}^{n}(x_i+y_i)}$
Hassanat	$\sum_{i=1}^{n} D(x_i,y_i)$ $=\begin{cases}1-\frac{1+\min(x_i,y_i)}{1+\max(x_i,y_i)}, & \min(x_i,y_i)\geq 0\\ 1-\frac{1+\min(x_i,y_i)+\lvert\min(x_i,y_i)\rvert}{1+\max(x_i,y_i)+\lvert\min(x_i,y_i)\rvert}, & \min(x_i,y_i)<0\end{cases}$

Abu Alfeilat HA, Hassanat ABA, Lasassmeh O, Tarawneh AS, Alhasanat MB, Eyal Salman HS, Prasath VBS. Effects of Distance Measure Choice on K-Nearest Neighbor Classifier Performance: A Review. Big Data. 2019 Dec;7(4):221-248. doi: 10.1089/big.2018.0175. Epub 2019 Aug 14. PMID: 31411491.

Distances with other data types

- **Hamming distance**: the number of symbols or positions of two strings at which their corresponding characters are different (> binary data).
- **Edit distance**: minimum number of operations required to transform one string into the other (> generalization of Hamming)

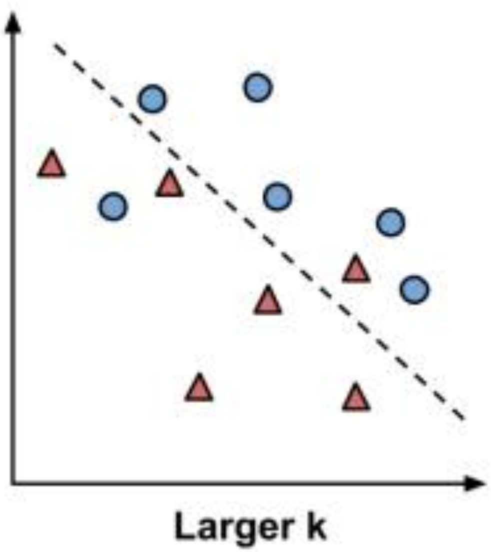


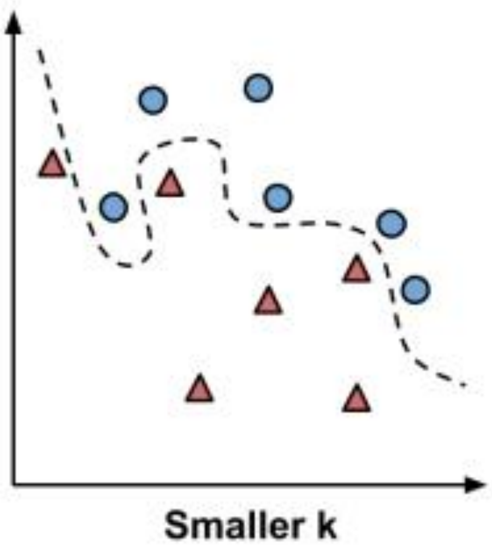
Hamming distances between binary digits [Wikipedia]

Calculation of Levenshtein distance between two strings [Wikipedia]

Choosing an appropriate k

- The balance between *underfitting* and *overfitting* the training data is a problem known as the *bias-variance tradeoff*.





Preparing data for use with k-NN

- Features are typically transformed to a standard range (normalized) prior to applying the k-NN algorithm.
- The rationale for this step is that the distance formula is dependent on how features are measured.
- In particular, if certain features have much larger values than others, the distance measurements will be strongly dominated by the larger values.

$$v' = new - min_A + R\left(\frac{v - min_A}{max_A - min_A}\right)$$

Min-max normalization

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

Z-score standardization

Preparing data for use with k-NN

- Qualitative features
  - Nominal:
    - Dummy coding
  - Ordinal:
    - Transform to numeric keeping ordering...

The k-NN algorithm

STRENGTHS

- Simple and effective
- Makes no assumptions about the underlying data distribution
- Fast training phase

WEAKNESSES

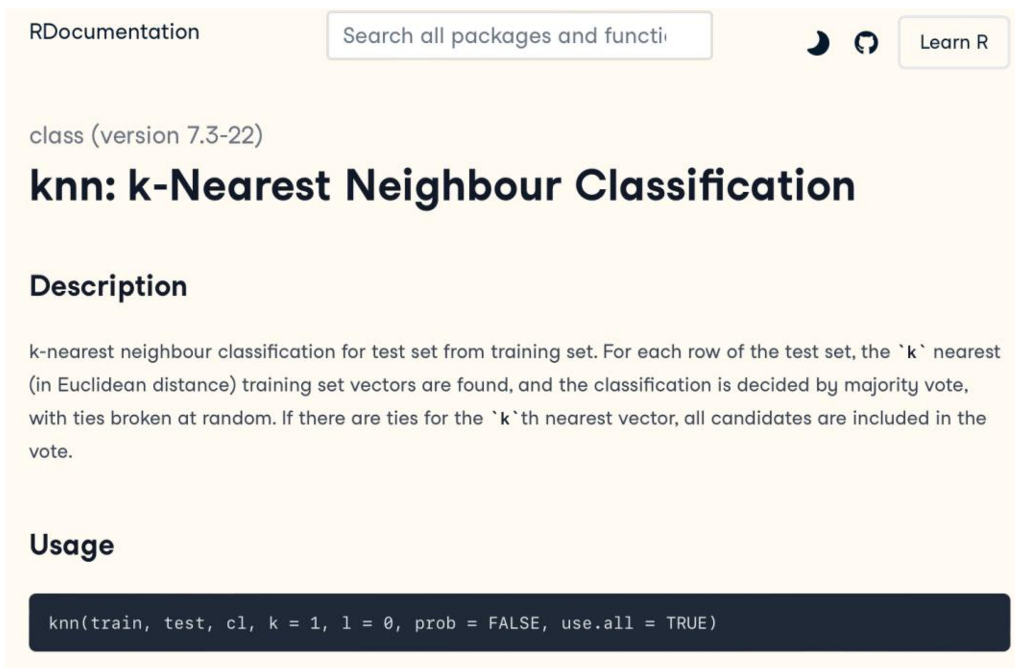
- Does not produce a model, which limits the ability to find novel insights in relationships among features
- Slow classification phase
- Requires a large amount of memory
- Qualitative features and missing data require additional processing

K-NN

R session

Exercise 1

- Revise the documentation of the function `knn` from the `class` library.


RDocumentation

Search all packages and functi

Learn R

class (version 7.3-22)

knn: k-Nearest Neighbour Classification

Description

k-nearest neighbour classification for test set from training set. For each row of the test set, the `k` nearest (in Euclidean distance) training set vectors are found, and the classification is decided by majority vote, with ties broken at random. If there are ties for the `k`th nearest vector, all candidates are included in the vote.

Usage

```
knn(train, test, cl, k = 1, l = 0, prob = FALSE, use.all = TRUE)
```

https://www.rdocumentation.org/packages/class/versions/7.3-22/topics/knn

Exercise 1

- Create a function `my_knn` that accepts any measure from the *philentropy* package and performs basic knn.
  - A possible function interface could be:

```
my_knn <- function(train, train_labels, test, k=1, metric="euclidean")
```

  - The function will output the predictions over the test set.

- Select two distance/similarity measures and apply the `my_knn` function to each of them with different k choices for the *"cancer"* data (see *cancer.qmd*).
- Do a comparison of the results (try using a plot for the accuracy values, e.g., barplot or Fig 2.17 before).