Chapter 6. Similarity and Nearest Neighbours

- *Basic concepts*: Similarity measures for objects described by data, using similarity for prediction
- *Exemplary techniques*: searching for similar entities, nearest neighbour methods.

Basic idea: If two things (people, products, companies) are similar in some ways, they often share other characteristics as well

Further readings: Provost and Fawcett (2013) Chapter 6 (1st half), James et al. (2013) Section 4.6.5.

Different business tasks involve reasoning from similar examples:

- Retrieve similar things directly
  IBM wants to find companies which are similar to their best business customers
  HP maintains many high-performance severs for clients, aided by a tool that, given a server configuration, retrieves information on other similarly configured servers
  Advertisers serve online ads to consumers who are similar to their current good customers

- Use similarity for classification and regression

- Use similarity for clustering
  Divide the entire customer base into several clusters such that the customers within each cluster are similar in some aspects

- Recommend similar products
  'Customers who bought X have also bought Y'
  'Customers with your browsing history have also looked at $\cdots$'

- Beyond business
  A doctor may reason about a new difficult case by recalling a similar case
  A lawyer often argues cases by citing legal precedents
  Chinese Medicine
  Case-based reasoning systems are built based on Artificial Intelligence

Similarity and Distance

Objects are represented by data. Typically each object has multiple attributes, i.e. represented by a vector. Therefore we may calculate

the distance between two vectors: the larger the distance is, the less similar the two objects are.

There are many distance measures for vectors $\mathbf{x} = (x_1, \ldots, x_p)^T, \mathbf{y} = (y_1, \cdots, y_p)^T \in R^p$.

- Euclidean (or L_2) distance: $\left((x_1 - y_1)^2 + \cdots + (x_p - y_p)^2\right)^{1/2}$
- L_1 distance: $|x_1 - y_1| + \cdots + |x_p - y_p|$
- L_∞ distance: $\max_j |x_j - y_j|$
- Weighted L_2 distance: $\left(w_1(x_1 - y_1)^2 + \cdots + w_p(x_p - y_p)^2\right)^{1/2}$, where $w_j \geq 0$ are a set of weights

- Correlation based distance: $1 - \rho(\mathbf{x}, \mathbf{y})$, where

$$\rho(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{p} (x_i - \bar{x})(y_i - \bar{y}) \Big/ \Big\{ \sum_{i=1}^{p} (x_i - \bar{x})^2 \sum_{j=1}^{p} (y_j - \bar{y})^2 \Big\}^{1/2}.$$

- Mahalanobis distance: Let $\mathbf{x}_i = (x_{i1}, \cdots, x_{ip})'$, $i = 1, \cdots, n$ be n vectors, the Mahalanobis distance between any pair $\mathbf{x}_i$ and $\mathbf{x}_j$ is

$$d_{ij} \equiv \Big\{ \sum_{k,l=1}^{p} (x_{ik} - x_{jk})(x_{il} - x_{jl}) a_{kl} \Big\}^{1/2},$$

where $\mathbf{A} = (a_{ij})$ is the inverse of the sample covariance matrix

$$\mathbf{A}^{-1} = \frac{1}{n-1} \sum_{j=1}^{n} (\mathbf{x}_j - \bar{\mathbf{x}})(\mathbf{x}_j - \bar{\mathbf{x}})^T, \quad \bar{\mathbf{x}} = \frac{1}{n} \sum_{j=1} \mathbf{x}_j.$$

Note. The Mahalanobis distance can be viewed as a distance of the normalized and decorrelated data: $\mathbf{y}_i = \mathbf{A}^{1/2} \mathbf{x}_i$, and

$$d_{ij} = \Big\{ \sum_{k=1}^{p} (y_{ik} - y_{jk})^2 \Big\}^{1/2}.$$

A credit card marketing problem: predict if a new customer will respond to a credit card offer based on how other, similar customers have responded.

Customer	Age	Income	Cards	Response	Distance to David
David	37	50	2	?	0
John	35	35	3	Yes	15.16
Rachael	22	50	2	No	15
Ruth	63	200	1	No	152.23
Jefferson	59	170	1	No	122
Norah	25	40	4	Yes	15.74

Note. Euclidean distance was used. For example, the distance between John and David is

$$15.16 = \sqrt{(35-37)^2 + (35-50)^2 + (3-2)^2}.$$

Rachael is most similar to David, and Ruth is most dissimilar to David.

Distance, as a (dis)similarity measure, is just a number: it has no units, no meaningful interpretation in general.

Distance is useful for comparing the similarity among different pairs: the relative sizes matter.

Distance measures should be used with care, especially the attributes (i.e. components of vectors) are inhomogeneous, contain some nominal variables, or on different scales.

Further consideration: using Mahalanobis distance or a weighted Euclidean distance (with heavier weight on, for example, number of cards)?

Example: Whiskey Analysis

Source: `http://adn.biol.umontreal.ca/~numericalecology/data/scotch.html`

For someone who loves *single malt Scotch whiskey*, it is important to be able to identify those similar to a particular single malt he really likes, among hundreds of different single malts.

Single malt Scotch whisky is one of the most revered spirits in the world. It has such scope for variation, it can offer complexity or simplicity, unbridled power or a subtle whisper. To legally be called a single malt Scotch, the whisky must be distilled at a single distillery in Scotland, in a copper pot still from nothing other than malted barley, yeast and water. It must then be aged in an oak cask for at least three years and a day, and be bottled at no less than 40% abv.

Suppose Foster likes *Bunnahabhain*, he likes to find other ones like that among all the many single malts.

We need to define a feature vector for each single malt such that the similar vectors (i.e. with small distances among them) represent whiskeys with similar taste.

Lapointe and Legendre (1994). A classification of pure malt Scotch whiskies. *Applied Statistics*, **43**, 237-257.

Jackson (1989) *Michael Jackson's Malt Whisky Companion: A Connoisseur's Guide to the Malt Whiskies of Scotland*: tasting notes for 109 different single malt Scotches.

The tasting notes are in the form of literary descriptions on 5 whiskey attributes: Color, Nose, Body, Palate, Finish.

One example: 'Appetizing aroma of peat smoke, almost incense-like, heather honey with a fruity softness'.

Question: How to turn those literary descriptions into numerical feature vectors?

Collecting different values for 5 attributes:

Color: yellow, very pale, pale, gold, pale gold, old gold, full gold, amber, etc (14 values)

Nose: aromatic, peaty, sweet, light, fresh, sea, dry, grassy, etc (12 values)

Body: soft, full, round, smooth, firm, medium, light, oily (8 values)

Palate: full, dry, sherry, big, sweet, fruity, clean, grassy, smoky, salty, etc (15 values)

Finish: full, dry, warm, light, smooth, clean, fruity, grassy, smoky, etc (19 values)

The values in blue are characteristics for Bunnahabhain. Multiple values are possible for each whiskey!

68 values in total: a 68×1 vector with components equal to 1 (present) or 0 (absent).

Calculate the Euclidean distances between Bunnahabhain and each of the other 108 single malts, the most similar ones are:

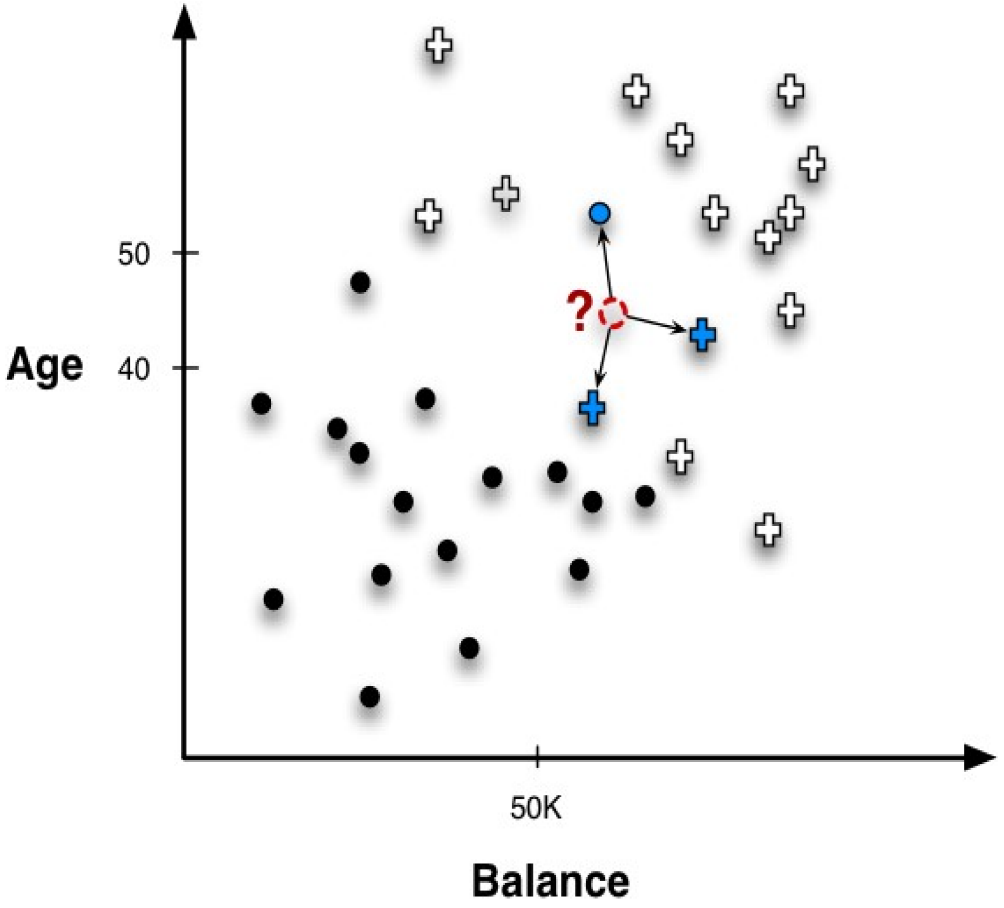
Whiskey	Distance	Description
Bunnahabhain	0	gold; firm,ned,light; sweet,fruit,clean; fresh,sea; full
Glenglassaugh	.643	gold; firm,light,smooth; sweet,grass; fresh,grass
Tullibardine	.647	gold; firm,med,smooth; sweet,fruit,full,grass, clean; sweet; big,aroma,sweet
Ardbeg	.667	sherry; firm,med,full,light; sweet; dry,peat, sea; salt
Bruichladdich	.667	pale; firm,light,smooth; dry,sweet,smoke,clean; light; full
Glenmorangie	.667	p.gold; med,oily,light; sweet,grass,spice; sweet,spicy,grass,sea,fresh; full, long

Nearest Neighbours (NN)

Similarity defines nearest neighbours for a individual: the individual with the smallest distance is the nearest neighbour. For any $k \geq 1$, the k-nearest neighbours are the k individuals. with the k smallest distances.

Using NN for prediction: predict the target value for the new individual by using the average (for regression) or the majority votes (for classification) of the known target values of its k-nearest neighbours among the training data.

Question: How to choose k?



NN-classification: The point labeled with ? is classified as + by 3-NN method, as the majority (i.e. 2 out of 3) of its neighbours are +.

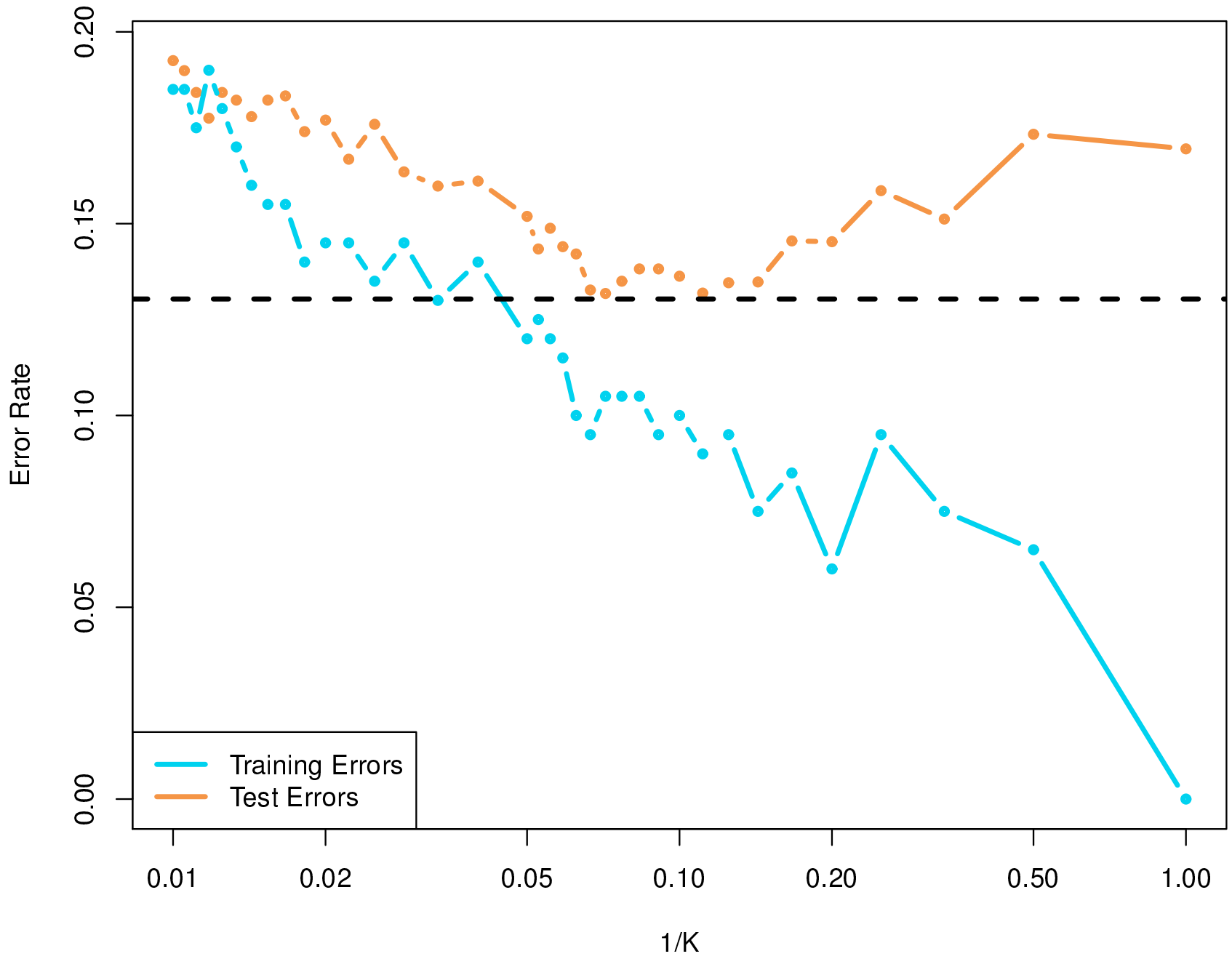
How many neighbours?

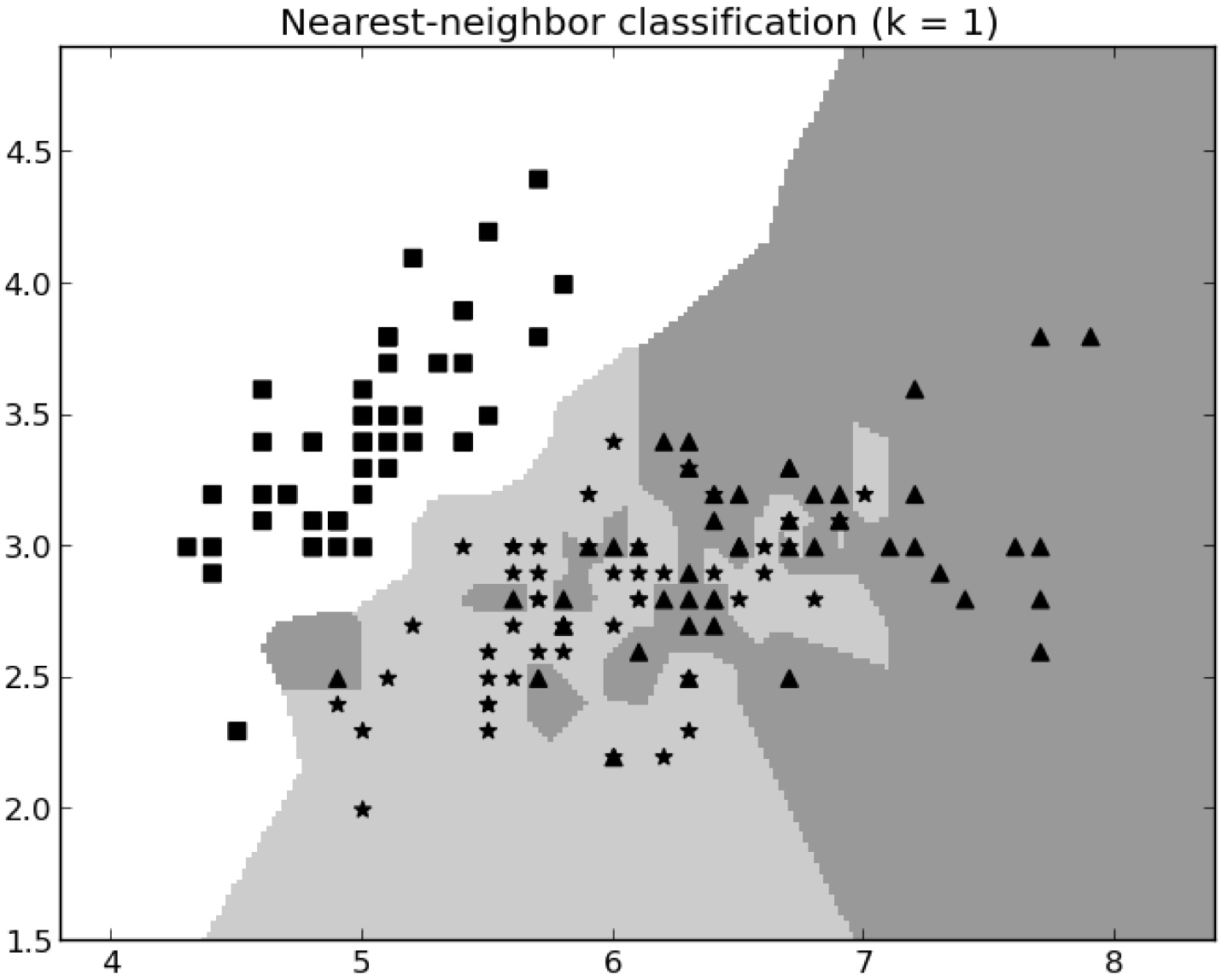
The k in k-NN method serves as a smooth parameter: the larger k is, the more smooth the method is.

When $k = n$, the implied regression model is a constant (i.e. sample mean), and the implied classification method puts all objects into one class.

$k = 1$ leads to overfitting, the complexity of the method is at its maximum,

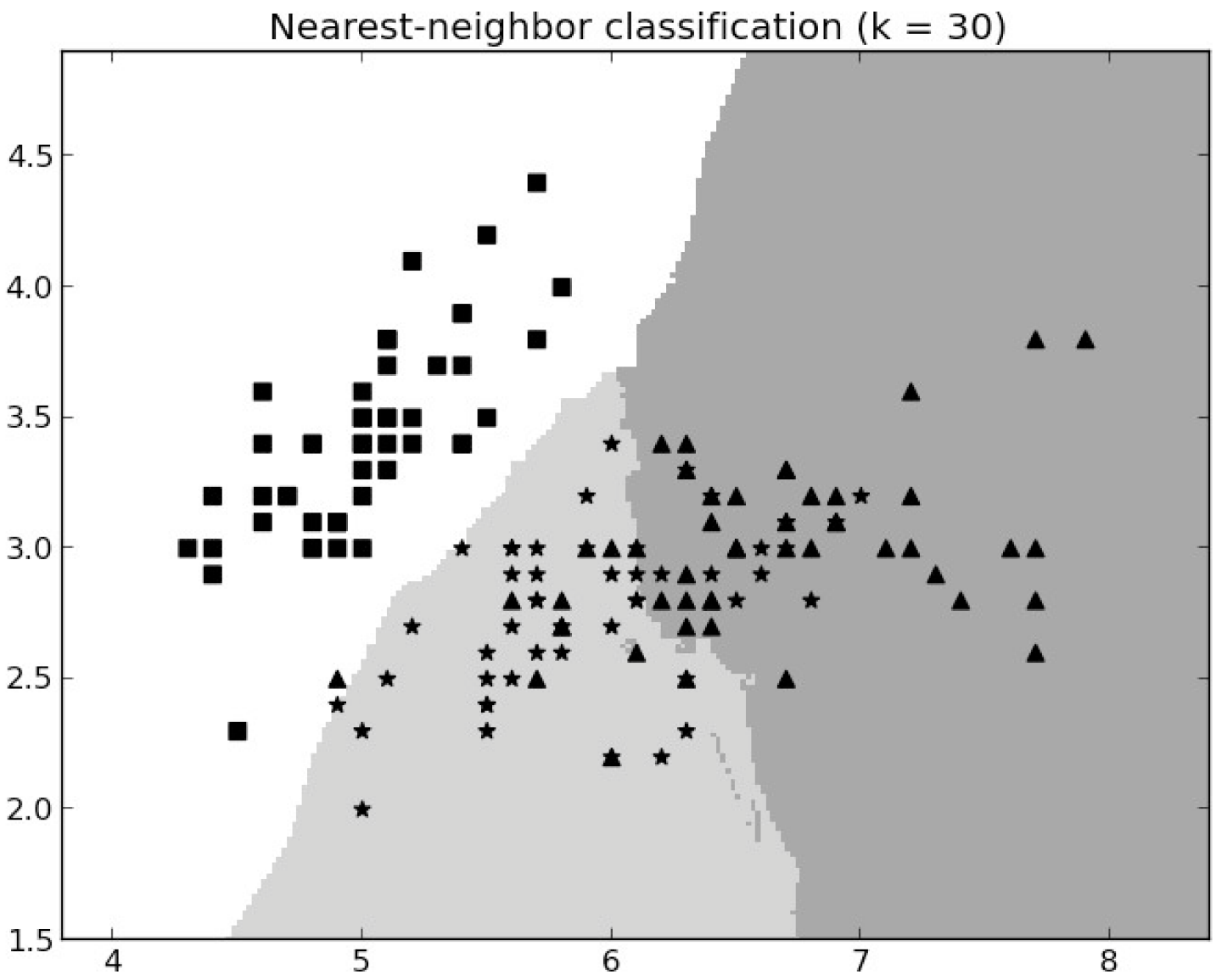
Choosing k by cross-validation or the other holdout methods.


0.20
0.15
0.10
0.05
0.00
Error Rate
Training Errors
Test Errors
0.01
0.02
0.05
0.10
0.20
0.50
1.00
1/K



3 class,

classification boundaries created by 1-NN



3 class,

classification boundaries created by 30-NN

A NN approach to making recommendations

Preparing recommendation for a new customer in an automatic manner may consist of three steps:

- Building a customer profile by getting the new customer to rate a selection of items such as movies, songs, or restaurants

- Comparing the new customer profile with the profiles of other customers using some measure of similarity

- Using some combination of the ratings of customers with similar profiles to predict the rating that the new customer would give to items he or she has not yet rated.

Building profiles

Sparseness of profiles: There are often far more items to be rated than any one person is likely to have experienced or be willing to rate. A user profile is a numeric vector consisting of, e.g. the digits between -5 (most negative) to 5 (most positive), while 0 stands for neutrality or no opinion.

On the other hand, forcing customers to rate a particular subset may miss interesting information because ratings of more obscure items may say more about the customer than ratings of common ones. A fondness for the Beatles is less revealing than a fondness for Mose Allison.

A reasonable approach is to have new customers rate a list of the twenty or so most frequently rated items (a list that might change over time) and then free them to rate as many additional items as they please.

Comparing profiles

Once a customer profile has been built, the next step is to measure its distance from other profiles. The most obvious approach would be to treat the profile vectors as geometric points and calculate the Euclidean distance between them, but many other distance measures have been tried. Some give higher weight to agreement when users give a positive rating especially when most users give negative ratings to most items. Still others apply statistical correlation tests to the ratings vectors.

Making Predictions

The final step is to use some combination of nearby profiles in order to come up with estimated ratings for the items that the customer has not rated. One approach is to take a weighted average where the weight is inversely proportional to the distance.

Issues with Nearest-Neighbour Methods

- *Justification*

  In some fields such as medicine or law, reasoning about similar cases is a natural way of making decision about a new case, a NN method may be a good fit.

  However a mortgage applicant may not be satisfied with the explanation: 'We decline your application because you remind us of the Smiths and the Mitchells, who both defaulted'.

  In contrast with a regression model, one may be able to say: 'all else being equal, if your income has been $ 20,000 higher, you would have been granted this mortgage.'

  Some careful and judicious presentation of NN based decision is useful.

Netflix uses a NN classification for their recommendations, explaining the recommendations with sentences like: 'The movie *Billy Elliot* was recommended based on your interest in *Amadeus, The Constant Gardener* and *Little Miss Sunshine*'.

- *Interpretation*

  It is difficult to explain what 'knowledge' has been used from the data in a NN method.

- *Dimensionality*

  Most practical problems have too many seemingly relevant attributes, i.e. vectors are very long. For a particular problem, many those attributes are irrelevant. In practice, either feature/variable selection method should be used (multi-fold CV) to select those to be used for calculating distance measures, or domain knowledge should be used to define an appropriate distance.

We consider a wholesale customers data set available at UCI Machine Learning Repository

`https://archive.ics.uci.edu/ml/datasets/wholesale+customers`

It is also available from the course Moodle as 'wholesaleCustomers.csv'. Download the data set and put it in your working directory.

```
> customers=read.csv("wholesaleCustomers.csv")
> dim(customers)
[1] 440   8
> View(customers)
```

It contains the info on 440 clients of a wholesale distributor: the annual spending on monetary units (m.u.) on 6 product categories.

Since the dataset is openly available, it has been analysed for quite a few times already. You can easily find them via google search.

Note. Google Dataset Search `https://toolbox.google.com/datasetsearch` can help you to find many datasets. Try searching for, eg., customers, churn, recommendation.

There are 8 variables/columns:

`Channel`: 1 - Horeca (Hotel/Restaurant/Cafe), 2 - Retail

`Region`: 1 - Lisbon, 2 - Oporto, 3 - Other regions

`Fresh`: annual spending (m.u.) on fresh products

`Milk`: annual spending on milk products

`Grocery`: annual spending on grocery products

`Detergents_paper`: annual spending on detergents and paper products

`Delicatessen`: annual spending on delicatessen products.

```
> summary(customers)
    Channel          Region          Fresh             Milk          Grocery
 Min.   :1.000   Min.   :1.000   Min.   :     3   Min.   :   55   Min.   :    3
 1st Qu.:1.000   1st Qu.:2.000   1st Qu.:  3128   1st Qu.: 1533   1st Qu.: 2153
 Median :1.000   Median :3.000   Median :  8504   Median : 3627   Median : 4756
 Mean   :1.323   Mean   :2.543   Mean   : 12000   Mean   : 5796   Mean   : 7951
 3rd Qu.:2.000   3rd Qu.:3.000   3rd Qu.: 16934   3rd Qu.: 7190   3rd Qu.:10656
 Max.   :2.000   Max.   :3.000   Max.   :112151   Max.   :73498   Max.   :92780
    Frozen        Detergents_Paper    Delicassen
 Min.   :   25.0   Min.   :    3.0   Min.   :    3.0
 1st Qu.:  742.2   1st Qu.:  256.8   1st Qu.:  408.2
```

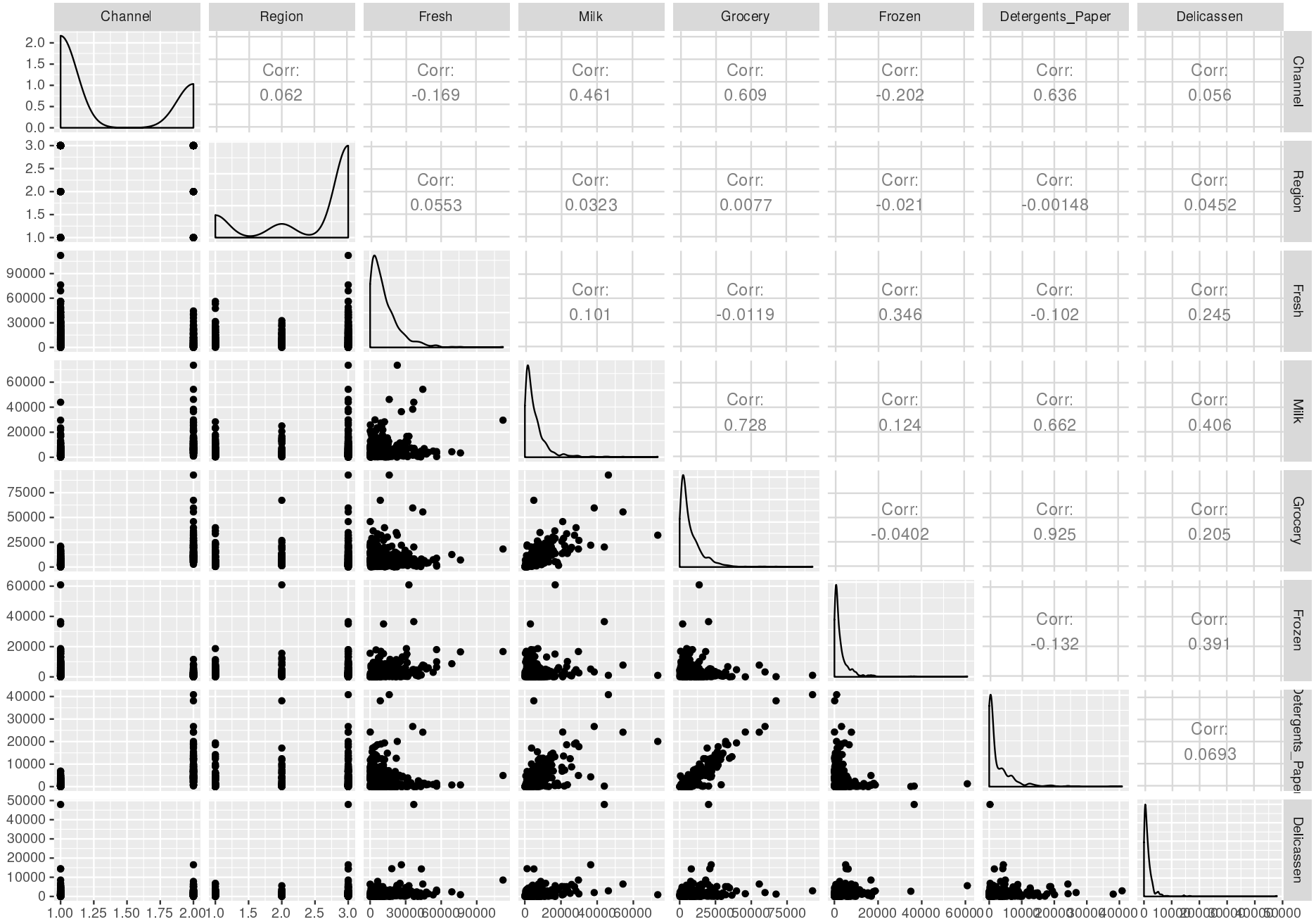
```
 Median : 1526.0   Median :  816.5   Median :  965.5
 Mean   : 3071.9   Mean   : 2881.5   Mean   : 1524.9
 3rd Qu.: 3554.2   3rd Qu.: 3922.0   3rd Qu.: 1820.2
 Max.   :60869.0   Max.   :40827.0   Max.   :47943.0
> library(GGally)
> ggpairs(customers)
```

No missing values, substantial variations in spending.

`Region` shows little correlation with all other variables

Only retail clients spend heavily on `Grocery` and `Detergents_paper`

Strong correlations between `Grocery` and `Detergents_paper`, `Milk` and `Grocery`


Channel
Region
Fresh
Milk
Grocery
Frozen
Detergents_Paper
Delicassen
Corr: 0.062
Corr: -0.169
Corr: 0.461
Corr: 0.609
Corr: -0.202
Corr: 0.636
Corr: 0.056
Corr: 0.0553
Corr: 0.0323
Corr: 0.0077
Corr: -0.021
Corr: -0.00148
Corr: 0.0452
Corr: 0.101
Corr: -0.0119
Corr: 0.346
Corr: -0.102
Corr: 0.245
Corr: 0.728
Corr: 0.124
Corr: 0.662
Corr: 0.406
Corr: -0.0402
Corr: 0.925
Corr: 0.205
Corr: -0.132
Corr: 0.391
Corr: 0.0693

Now how we define the distances between pair clients? Here are some possibilities:

1. Euclidean, L_1 or maximum component distance:

```
> dist_euc=dist(customers, method="euclidean")
 # Change euclidan to maximum for the maximum component, to manhattan
 # for $L_1$ distance, and to canberra for
 # $\sum_i |x_i - y_i|/|x_i + y_i|$
> summary(dist_euc)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
   278.9  10083.6  16302.6  20848.7  25741.6 128968.4
> length(dist_euc)
[1] 96580  # = 440*439/2
> dist_euc=as.matrix(dist(customers, method="euclidean")) #  distance matrix
> dim(dist_euc)
[1] 440 440
> sort(dist_euc[2,])[1:4] #  rearrange the componemts  in ascending order
       2      245      397      165
   0.000 2612.974 3499.196 3509.257
# The 3 nearest neighbours of Row 2 are Rows 245, 397, 165
```

The contribution of `channel, Region` in the above distance is negligible!

2. absolute difference in `Channel` + Mahalanobis distance of 6 spending variables

```
> attach(customers)
> D1=outer(Channel, Channel, "-")  # D1[i,j] = Channel[i]-Channel[j]
> dim(D1)
[1] 440 440
> customer6=customers[,3:8]
> S=var(customer6)
> S
                     Fresh     Milk  Grocery   Frozen Detergents_Paper Delicassen
Fresh            159954927  9381789 -1424713 21236655       -6147825.7  8727310.0
Milk               9381789 54469967 51083186  4442612       23288343.5  8457924.8
Grocery           -1424713 51083186 90310104 -1854282       41895189.7  5507291.3
Frozen            21236655  4442612 -1854282 23567853       -3044324.9  5352341.8
Detergents_Paper  -6147826 23288343 41895190 -3044325       22732436.0   931680.7
Delicassen         8727310  8457925  5507291  5352342         931680.7  7952997.5
> tt=eigen(S, symmetric=T) # perform eigen-analysis for matrix S
> d=tt$values  # eigenvalues
> G=tt$vectors # 6x6 matrix with eigenvectors as columns
> S2=G%*%diag(1/sqrt(d))%*%t(G)  # S2 = S^{-1/2}
> customer6N=as.matrix(customer6)%*%S2  # Normalize the columns of customer6
> var(customer6N)
              [,1]          [,2]          [,3]          [,4]          [,5]          [,6]
[1,]  1.0000e+00 -3.0375e-17 -2.7856e-16  1.4898e-16 -2.8542e-16  6.8510e-17
[2,] -3.0375e-17  1.0000e+00  2.8102e-16  2.2774e-16  4.3874e-16  9.1557e-16
```

```
[3,] -2.7856e-16  2.8102e-16  1.0000e+00 -2.3778e-16  2.8542e-16 -4.6251e-16
[4,]  1.4898e-16  2.2774e-16 -2.3778e-16  1.0000e+00  2.4173e-16  5.4908e-16
[5,] -2.8542e-16  4.3874e-16  2.8542e-16  2.4173e-16  1.0000e+00  2.4843e-16
[6,]  6.8510e-17  9.1557e-16 -4.6251e-16  5.4908e-16  2.4843e-16  1.0000e+00
> dist_maha = abs(D1) + as.matrix(dist(customer6N, method="euclidean"))
> sort(dist_maha[2,])[1:4]
        2       109       397        83
0.0000000 0.5688088 0.5721256 0.6339615
```

Now the 3 nearest neighbours of Row 2 are Rows 109, 397 and 83.

The above calculation is based on the fact that a Mahalanobis distance is the Euclidean distance and normalized vectors.

We can use `k*abs(D1) + as.matrix(dist(customer6N, method="euclidean"))` instead, where constant $k > 0$ balances the relative importance of the two terms. We may even consider to choose k according to some appropriate criterion.

3. absolute difference in `Channel` + (1 - correlation based on 6 spending variables)

```
> dist_cor=abs(D1)+1-cor(t(customer6))
# cor calculates the correlations btw columns, hence tranpose t(customer6)
> sort(dist_cor[2,])[1:4]
[1] 0.00000000 0.01309468 0.04160087 0.04768788
> sort.int(dist_cor[2,], index.return=T)$ix[1:4]  # check ?sort and ?sort.int
[1]   2  48  95 165
> sort.int(dist_cor[2,], index.return=T)$x[1:4]
[1] 0.00000000 0.01309468 0.04160087 0.04768788
```

Now the 3 nearest neighbours of Row 2 are Rows 48, 95, 165.