Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Business Analytics using Data Mining

BU7143

Dr. Nicholas P. Danks
Business Analytics
nicholas.danks@tcd.ie

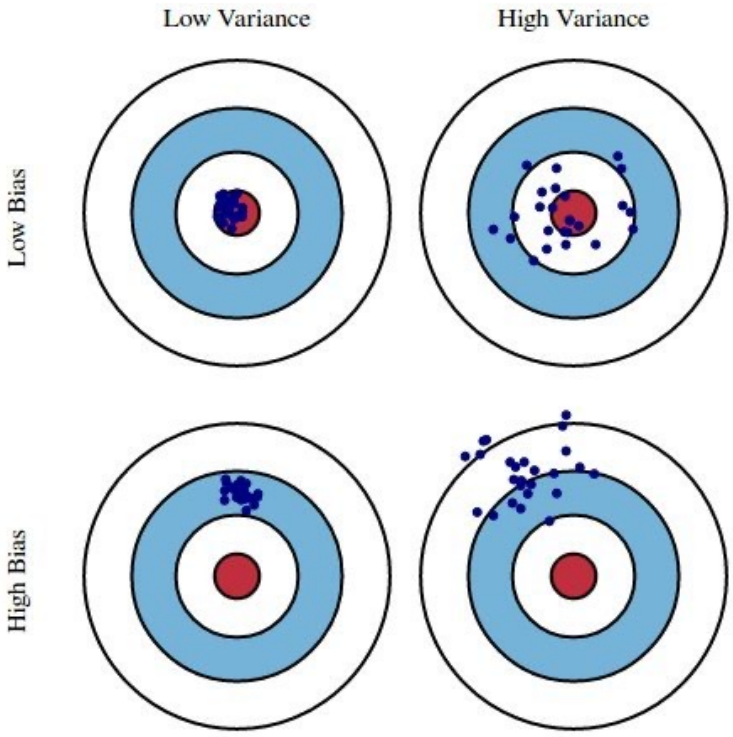
Overview of Today's Session

1. Performance Evaluation
2. K-Nearest Neighbours
3. Logistic Regression

Part I
Performance Evaluation

Why Evaluate?

- Multiple methods are available to classify or predict
- For each method, multiple choices are available for settings
- To choose best model, need to assess each model's performance



There are many ways to describe inaccuracy! What matters most?

Measuring Predictive error

- We want to evaluate how model predicts **new data**,
- NOT how well it fits the data it was trained on.
- Key component of most measures is difference between actual *y* and predicted *y* ("error")

Actual value (y) is observed
Fitted value ($\hat{y}$) is in-sample
Predicted value ($\hat{y}$) is out-sample

$$e = y - \hat{y}$$

- **MAE or MAD**: Mean absolute error (deviation)
  Gives an idea of the magnitude of errors
- **Average error**
  Gives an idea of systematic over- or under-prediction
- **MAPE**: Mean absolute percentage error
- **RMSE** (root-mean-squared-error): Square the errors, find their average, take the square root

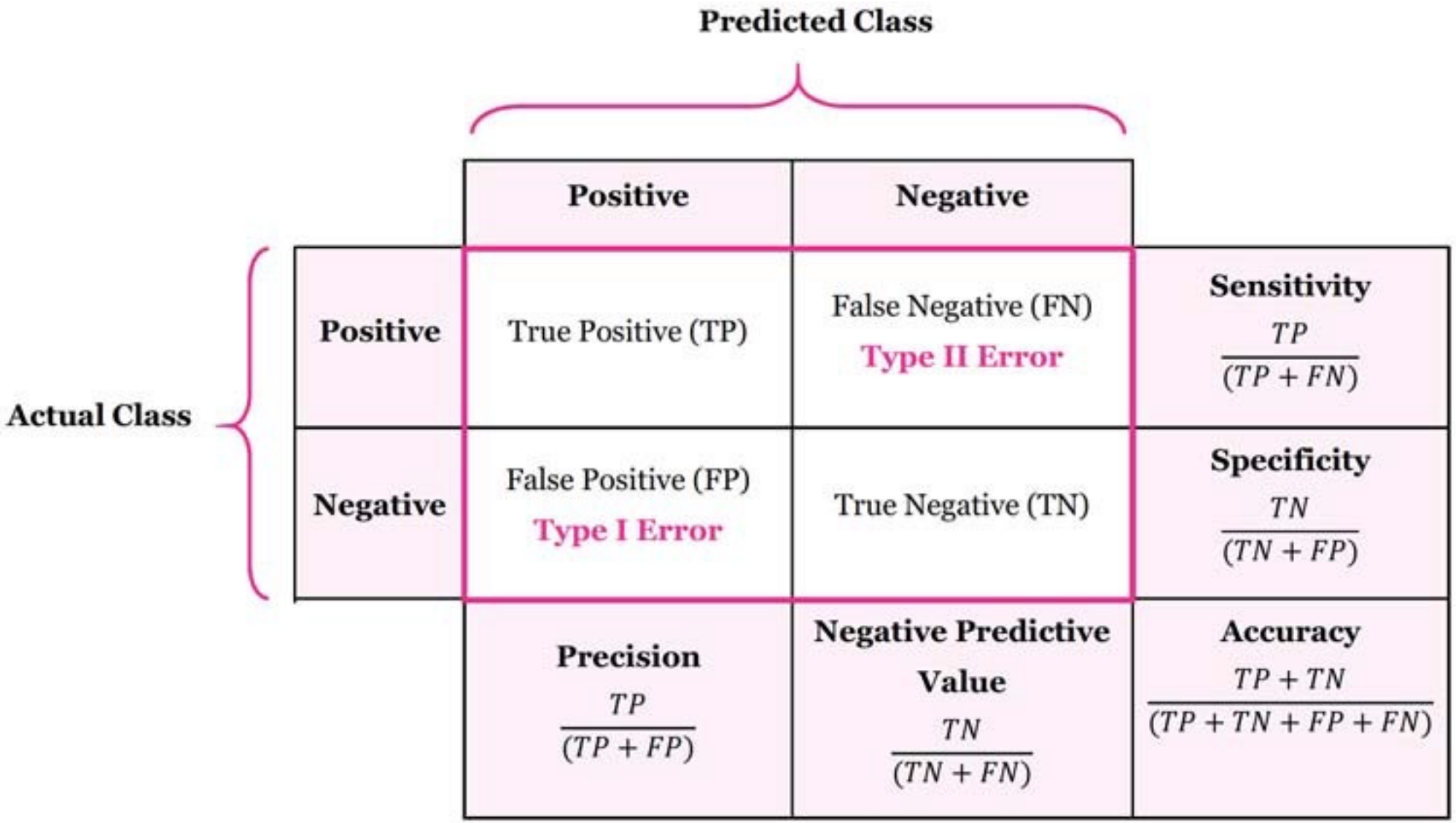
$$\text{MAE} = \frac{\sum |e|}{n}$$

$$\text{ME} = \frac{\sum e}{n}$$

$$\text{MAPE} = \frac{100}{n} \sum \left| \frac{e}{y} \right|$$

$$\text{RMSE} = \sqrt{\frac{\sum e^2}{n}}$$

Classification Matrix / Confusion Matrix



Misclassification error

- Error = classifying a record as belonging to one class when it belongs to another class.
- Error rate = percent misclassified

Predicted ($\hat{Y}$) \ Actual (Y)	0	1
0	2689	85
1	25	201

$$e = \frac{25+85}{3000} = 3.6\%$$

Naïve Rule

- Classify all records as belonging to the most prevalent class
- Often used as benchmark: we hope to do better than that
- Exception: when goal is to identify high-value but rare outcomes, we may do well by doing worse than the naïve rule (see "lift" – later)

	Actual O	Actual 1
Pred 0	2714	286
Pred 1	0	0

$$e = \frac{286}{3000} = 9.5\%$$

Our best guess is the average!

Cutoff for classification

- Most DM algorithms classify via a 2-step process:
- For each record,
  - Compute **probability of belonging to class "1"**
  - Compare to cutoff value, and classify accordingly
- Default cutoff value is 0.50
  - If >= 0.50, classify as "1"
  - If < 0.50, classify as "0"
- Can use different cutoff values
- Typically, error rate is lowest for cutoff = 0.50

Cutoff Table

Actual Class	Prob. of "1"	Actual Class	Prob. of "1"
1	0.996	1	0.506
1	0.988	0	0.471
1	0.984	0	0.337
1	0.980	1	0.218
1	0.948	0	0.199
1	0.889	0	0.149
1	0.848	0	0.048
0	0.762	0	0.038
1	0.707	0	0.025
1	0.681	0	0.022
1	0.656	0	0.016
0	0.622	0	0.004

Classification threshold is another parameter to be tuned!

Confusion Matrix for Different Cutoffs

Function `confusionMatrix` requires library `caret`

```
## cutoff = 0.5
> confusionMatrix(ifelse(owner.df$Probability>0.5,
# note: "reference" = "actual"
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner       10     1
  owner           2    11

               Accuracy : 0.875
```

```
## cutoff = 0.25
> confusionMatrix(ifelse(owner.df$Probability>0.25,
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner        8     1
  owner           4    11

               Accuracy : 0.7916667
```

```
## cutoff = 0.75
> confusionMatrix(ifelse(owner.df$Probability>0.75,
Confusion Matrix and Statistics

          Reference
Prediction nonowner owner
  nonowner       11     5
  owner           1     7

               Accuracy : 0.75
```

When One Class is More Important

In many cases it is more important to identify members of one class

- Tax fraud
- Credit default
- Response to promotional offer
- Detecting electronic network intrusion
- Predicting delayed flights

In such cases, we are willing to tolerate greater overall error, in return for better identifying the important class for further attention

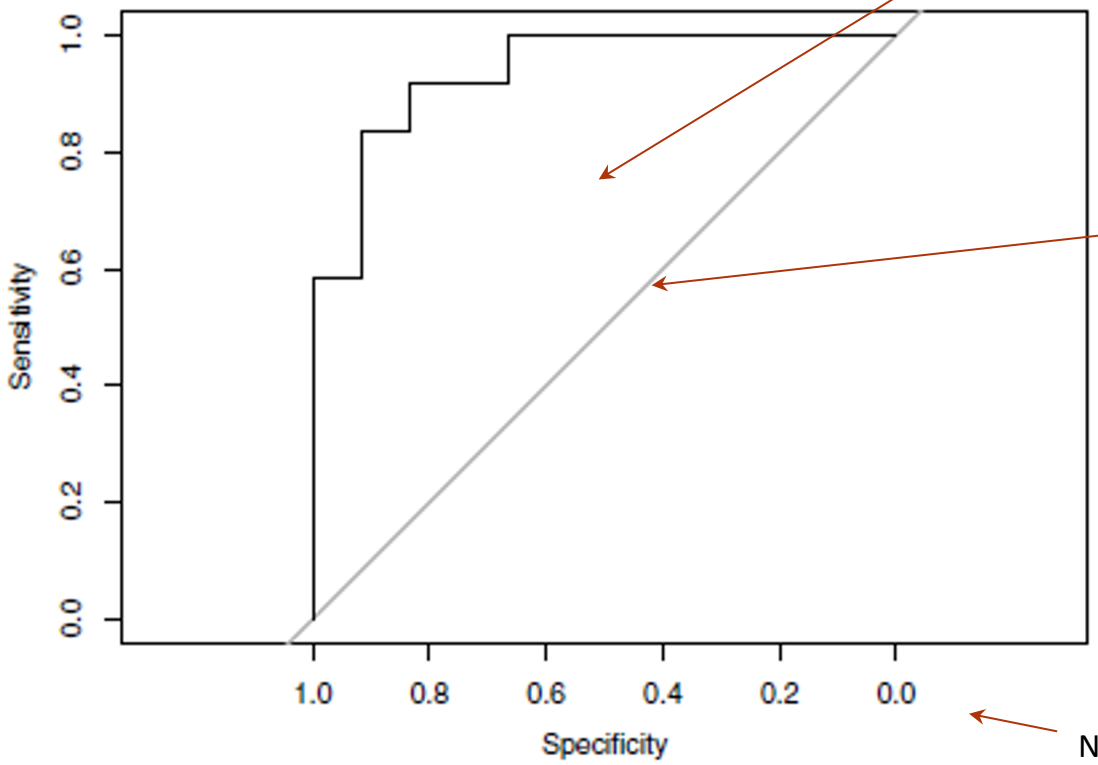
If "C_1" is the important class,

Sensitivity (also called "recall) = % of "C_1" class correctly classified

Specificity = % of "C_0" class correctly classified

Precision= % of predicted "C_1's" that are actually"C_1's"

ROC Curve (library pROC)



Area under the ROC curve ("AUC") is a useful metric

Diagonal reflects random classification on a sliding probability, from 0 to 1, of labeling a record as a "1"

Note that x-axis is in reverse order; alternative is to plot "1-specificity" in ascending order

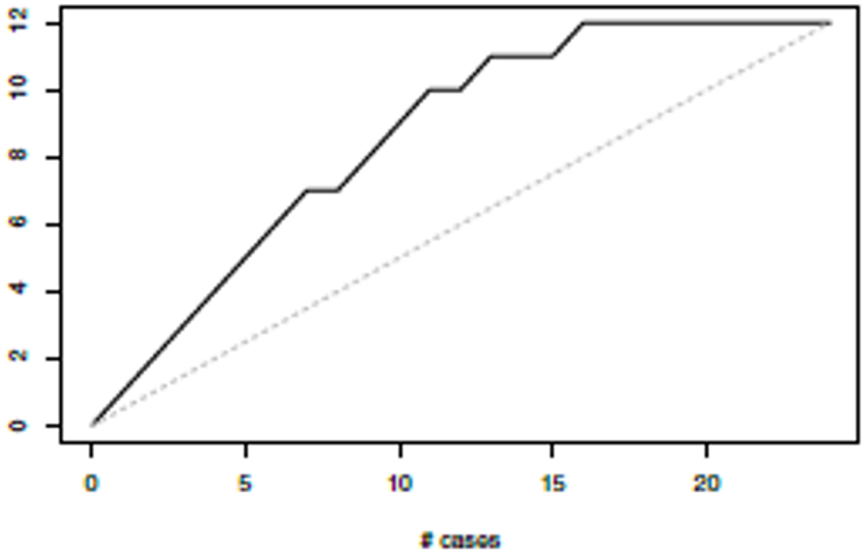
If "C_1" is the important class,

Sensitivity (also called "recall) = % of "C_1" class correctly classified

Specificity = % of "C_0" class correctly classified

Lift (“gains”): Goal

- Evaluates how well a model identifies the most important class
- Helps evaluate, e.g.,
  - How many tax records to examine
  - How many loans to grant
  - How many customers to mail offer to
- Compare performance of DM model to “no model, pick randomly”
- Measures ability of DM model to identify the important class, relative to the average prevalence of the class
- Charts give explicit assessment of results over a large number of cutoffs



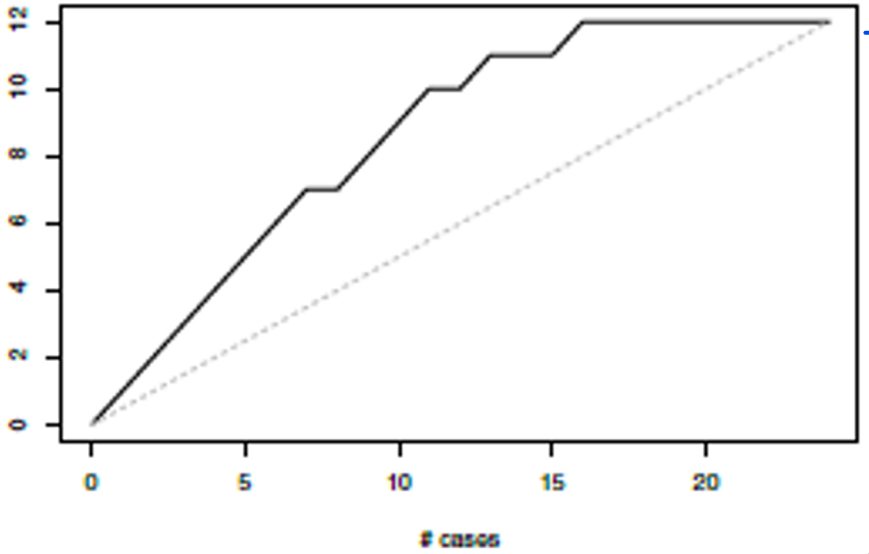
Lift and Decile Charts: How to Use

Sort records by predicted probability of belonging to the important class ("1's")

Move down the list, noting actual class

As you go, compare the number of actual 1's to the number of 1's you would expect with no model

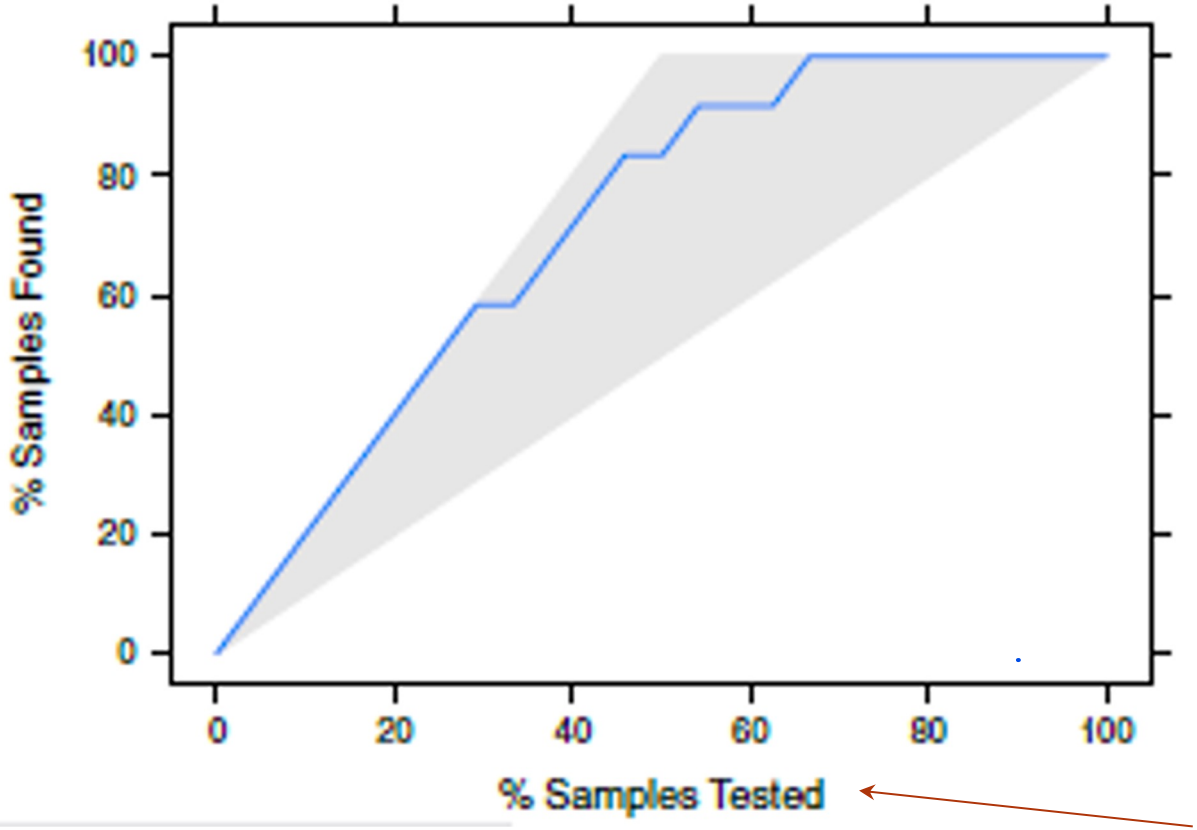
- In lift chart: compare step function to straight line
- In decile chart compare to ratio of 1



After examining (e.g.,) 10 cases (x-axis), 9 owners (y-axis) have been correctly identified

Lift Chart

using %



"Sample" = case

After examining (e.g.,) 40% = 10 of the cases (x-axis), 75% of the owners (y-axis) have been correctly identified

Decile Chart

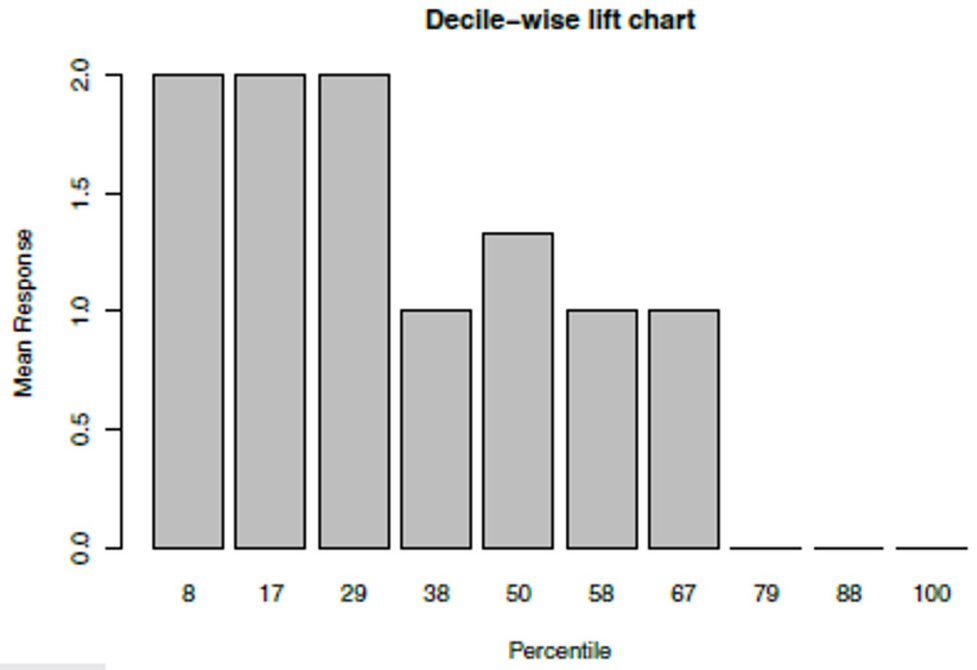


FIGURE 5.7 DECILE LIFT CHART

In “most probable” (top) decile, model is twice as likely to identify the important class compared to avg. prevalence. Percentiles do not match deciles exactly due to small sample of discrete data, with multiple records sharing same decile boundary.

Asymmetric Costs

Introducing Costs & Benefits

Suppose:

Profit from a "1" is $10

Cost of sending offer is $1

Then:

Under naïve rule, all are classified as "0", so no offers are sent: no cost, no profit

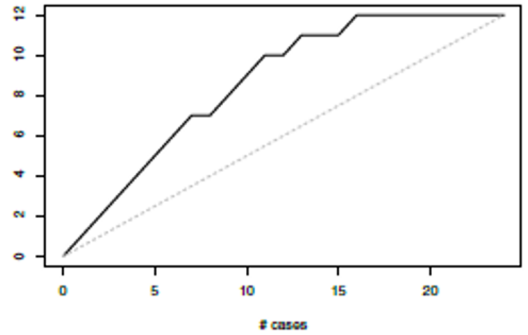
Under DM predictions, 28 offers are sent.

- 8 respond with profit of $10 each
- 20 fail to respond, cost $1 each
- 972 receive nothing (0 cost, 0 profit)

Net profit = $60

Profit Matrix

	Actual 0	Actual 1
Predicted 0	$0	$0
Predicted 1	($20)	$80



Adding costs to the mix, as above, does not change the actual classifications

Better: Use the lift curve and change the cutoff value for "1" to maximize profit

Generalize to Cost Ratio

Sometimes actual costs and benefits are hard to estimate

- Need to express everything in terms of costs (i.e., cost of misclassification per record)
- Goal is to minimize the average cost per record

A good practical substitute for individual costs is the **ratio** of misclassification costs (e,g,, "misclassifying fraudulent firms is 5 times worse than misclassifying solvent firms")

Minimizing Cost Ratio

q_1 = cost of misclassifying an actual "1",

q_0 = cost of misclassifying an actual "0"

Minimizing the **cost ratio** q_1/q_0 is identical to minimizing the average cost per record

Rare Cases

Asymmetric costs/benefits typically go hand in hand with presence of rare but important class

- Responder to mailing
- Someone who commits fraud
- Debt defaulter

- Often we oversample rare cases to give model more information to work with

- Typically use 50% “1” and 50% “0” for training

Summary

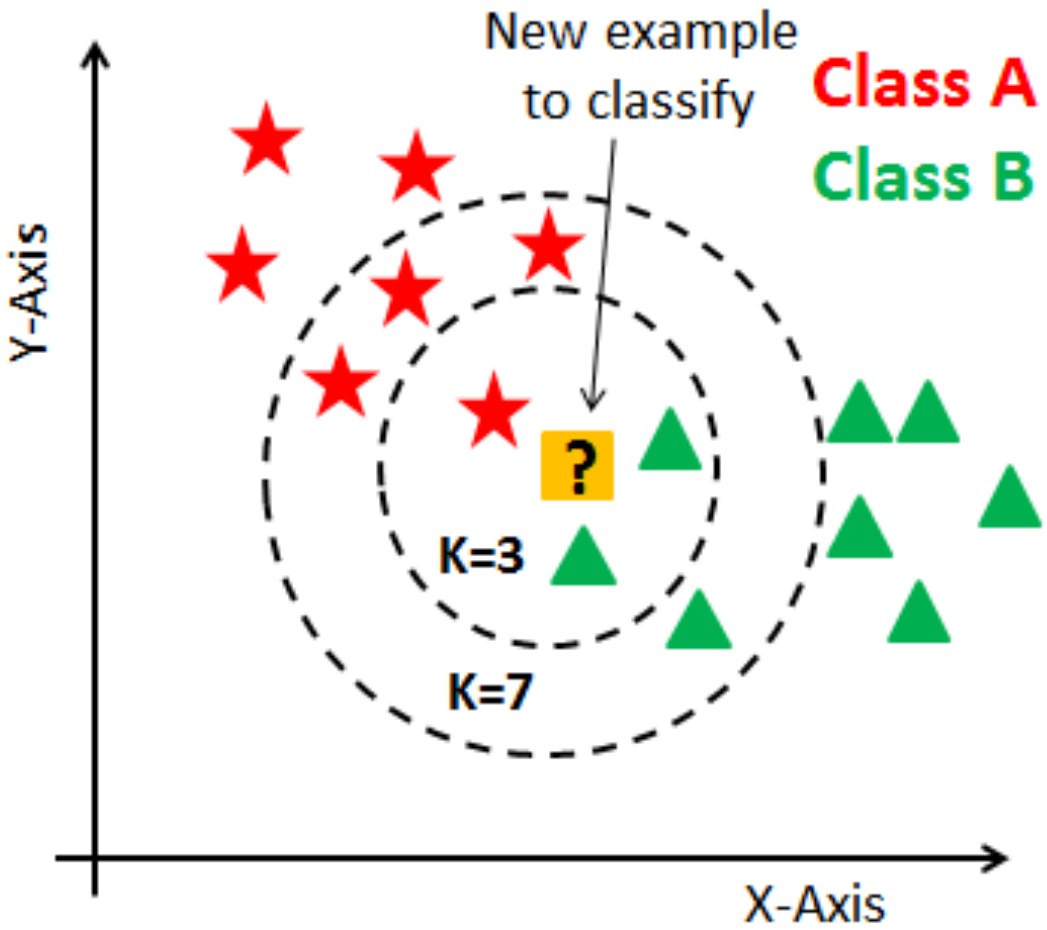
- Evaluation metrics are important for comparing across DM models, for choosing the right configuration of a specific DM model, and for comparing to the baseline (“no model”)
- Major metrics: confusion matrix, error rate, predictive error
- Other metrics when
  - one class is more important
  - asymmetric costs
- When important class is rare, use oversampling
- In all cases, metrics computed from validation data

Part II
K-NN

Characteristics of K-NN

What does ***"near"*** mean?

- Data-driven, not model-driven
- Makes no assumptions about the data



Basic Idea

- For a given record to be classified, identify nearby records
- "Near" means records with similar predictor values $X_1, X_2, \ldots X_p$
- Classify the record as whatever the predominant class is among the nearby records (the "neighbors")

KNN is democratic! The class is determined by popular vote.

How to measure “nearby”?

A lot more “distances” than you think:
- Manhattan
- Euclidean
…

https://en.wikipedia.org/wiki/Distance

The most popular distance measure is **Euclidean distance**

$$\sqrt{(x_1 - u_1)^2 + (x_2 - u_2)^2 + \cdots + (x_p - u_p)^2}$$

- Typically, predictor variables are first normalized (= standardized) to put them on comparable scales
- Use preProcess() from caret package to normalize
- Otherwise, variables with large scales dominate

Don’t forget about scale!!

Choosing k

- *K* is the number of nearby neighbors to be used to classify the new record
  - *K*=1 means use the single nearest record
  - *K*=5 means use the 5 nearest records
- Typically choose that value of *k* which has lowest error rate in validation data

Low *k* vs. High *k*

- Low values of *k* (1, 3, …) capture local structure in data (but also noise)
- High values of *k* provide more smoothing, less noise, but may miss local structure

  Note: the extreme case of k = n (i.e., the entire data set) is the same as the "naïve rule" (classify all records according to majority class)

Example: Riding Mowers

Data: 24 households classified as owning or not owning riding mowers

Predictors: Income, Lot Size

Income	Lot_Size	Ownership
60.0	18.4	owner
85.5	16.8	owner
64.8	21.6	owner
61.5	20.8	owner
87.0	23.6	owner
110.1	19.2	owner
108.0	17.6	owner
82.8	22.4	owner
69.0	20.0	owner
93.0	20.8	owner
51.0	22.0	owner
81.0	20.0	owner
75.0	19.6	non-owner
52.8	20.8	non-owner
64.8	17.2	non-owner
43.2	20.4	non-owner
84.0	17.6	non-owner
49.2	17.6	non-owner
59.4	16.0	non-owner
66.0	18.4	non-owner
47.4	16.4	non-owner
33.0	18.8	non-owner
51.0	14.0	non-owner
63.0	14.8	non-owner

Finding nearest neighbors in R

- Library FNN provides a list of neighbors
- Library `class` allows numerical output
- See Table 7.2 for code using knn from FNN library
- compares each record from validation* (or test) set to k nearest records in training
- Use library `caret` to get accuracy of different values of k, applied to validation data (see next slide for code)

Output

```
> accuracy.df
    k accuracy
1   1      0.7
2   2      0.7
3   3      0.8
4   4      0.9
5   5      0.8
6   6      0.9
7   7      0.9
8   8      1.0
9   9      0.9
10 10      0.9
11 11      0.9
12 12      0.8
13 13      0.4
14 14      0.4
```

- Even is possible for ties to occur
- R breaks ties randomly

Why is your result different even though we set.seed()?

Riding Mower Example

1. Partition Data
2. Visualize Data
3. Normalize Data
4. Select the value of k
5. Train the final model
6. Generate the prediction

KNN can yield a prediction as well as a classification. It is a VERY popular and powerful method.

New case:
Income: 60; Lot: 20

Can you guess the new case?

Income	Lot_Size	Ownership
60.0	18.4	owner
85.5	16.8	owner
64.8	21.6	owner
61.5	20.8	owner
87.0	23.6	owner
110.1	19.2	owner
108.0	17.6	owner
82.8	22.4	owner
69.0	20.0	owner
93.0	20.8	owner
51.0	22.0	owner
81.0	20.0	owner
75.0	19.6	non-owner
52.8	20.8	non-owner
64.8	17.2	non-owner
43.2	20.4	non-owner
84.0	17.6	non-owner
49.2	17.6	non-owner
59.4	16.0	non-owner
66.0	18.4	non-owner
47.4	16.4	non-owner
33.0	18.8	non-owner
51.0	14.0	non-owner
63.0	14.8	non-owner

Advantages

- Simple
- **No assumptions** required about Normal distribution, etc.
- Effective at capturing **complex interactions** among variables without having to define a statistical model

Shortcomings

- Required **size** of training set increases exponentially with # of predictors, *p*
  - This is because expected distance to nearest neighbor increases with *p* (with large vector of predictors, all records end up "far away" from each other)
- In a large training set, it takes **a long time** to find distances to all the neighbors and then identify the nearest one(s)
- These constitute "**curse of dimensionality**"

Dealing with the Curse

- Reduce dimension of predictors (e.g., with PCA)
- Computational shortcuts that settle for "almost nearest neighbors"

Part III
Logistic Regression

Logistic Regression

- Extension of linear regression where outcome variable is categorical
- Widely used, particularly model is useful to explain (=*profiling*) or to predict
- We focus on binary classification
  - i.e. Y=0 or Y=1

Goal: Find a function of the predictor variables that relates them to a 0/1 outcome

- Instead of Y as outcome variable (like in linear regression), we use a function of Y called the ***logit***
- Logit can be modeled as a linear function of the predictors
- The logit can be mapped back to a probability, which, in turn, can be mapped to a class

Step 1: Logistic Response Function

p = probability of belonging to class 1

Need to relate *p* to predictors with a function that guarantees $0 \leq p \leq 1$

Standard linear function (as shown below) does not:

$$p = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \beta_q x_q$$

The Fix: use ***logistic response function***

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \beta_q x_q)}}$$

Step 2: The Odds

The odds of an event are defined as:

$$Odds = \frac{p}{1-p}$$ ← p = probability of event

Or, given the odds of an event, the probability of the event can be computed by:

$$p = \frac{Odds}{1+Odds}$$

We can also relate the Odds to the predictors:

$$Odds = e^{\beta_0+\beta_1 x_1+\beta_2 x_2+\square+\beta_q x_q}$$

Step 3: Take log on both sides

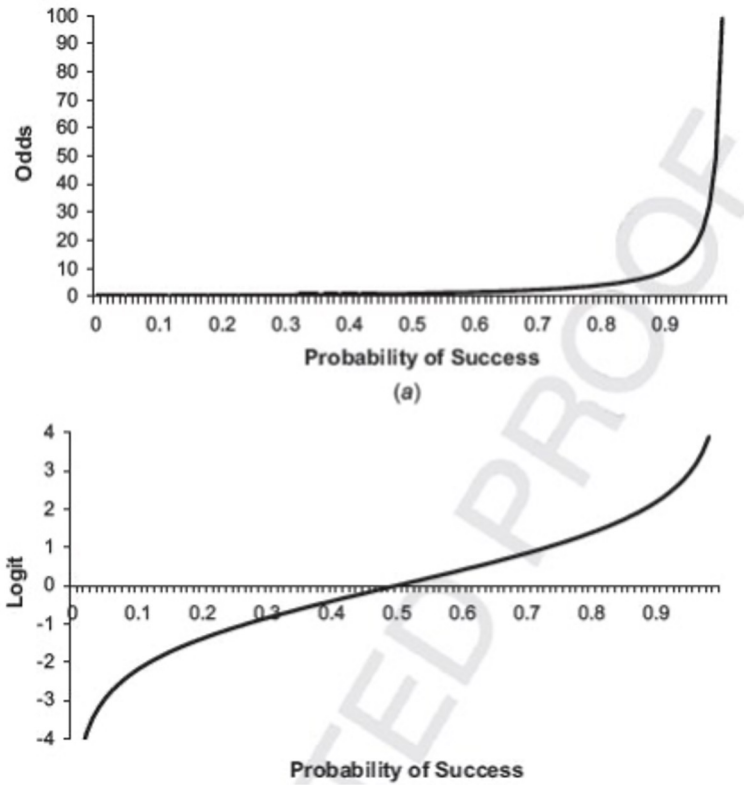
This gives us the logit:

$$\log(Odds) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \square + \beta_q x_q$$

log(Odds) = logit

So, the logit is a linear function of predictors x_1, x_2, …

- Takes values from -infinity to +infinity



Logistic Regression Example

Outcome variable: accept bank loan (0/1)

Predictors: Demographic info, and info about their bank relationship

Factor lets R know it is categorical

```
bank.df <- read.csv("UniversalBank.csv")
bank.df <- bank.df[ , -c(1, 5)]  # Drop ID and zip code columns.
# treat Education as categorical (R will create dummy variables)
bank.df$Education <- factor(bank.df$Education, levels = c(1, 2, 3),
    labels = c("Undergrad", "Graduate", "Advanced/Professional"))

# partition data
set.seed(2)
train.index <- sample(c(1:dim(bank.df)[1]), dim(bank.df)[1]*0.6)
train.df <- bank.df[train.index, ]
valid.df <- bank.df[-train.index, ]
```

Training partition of 60%

Thought Experiment: Single Predictor Model

Modeling loan acceptance on income (*x*)

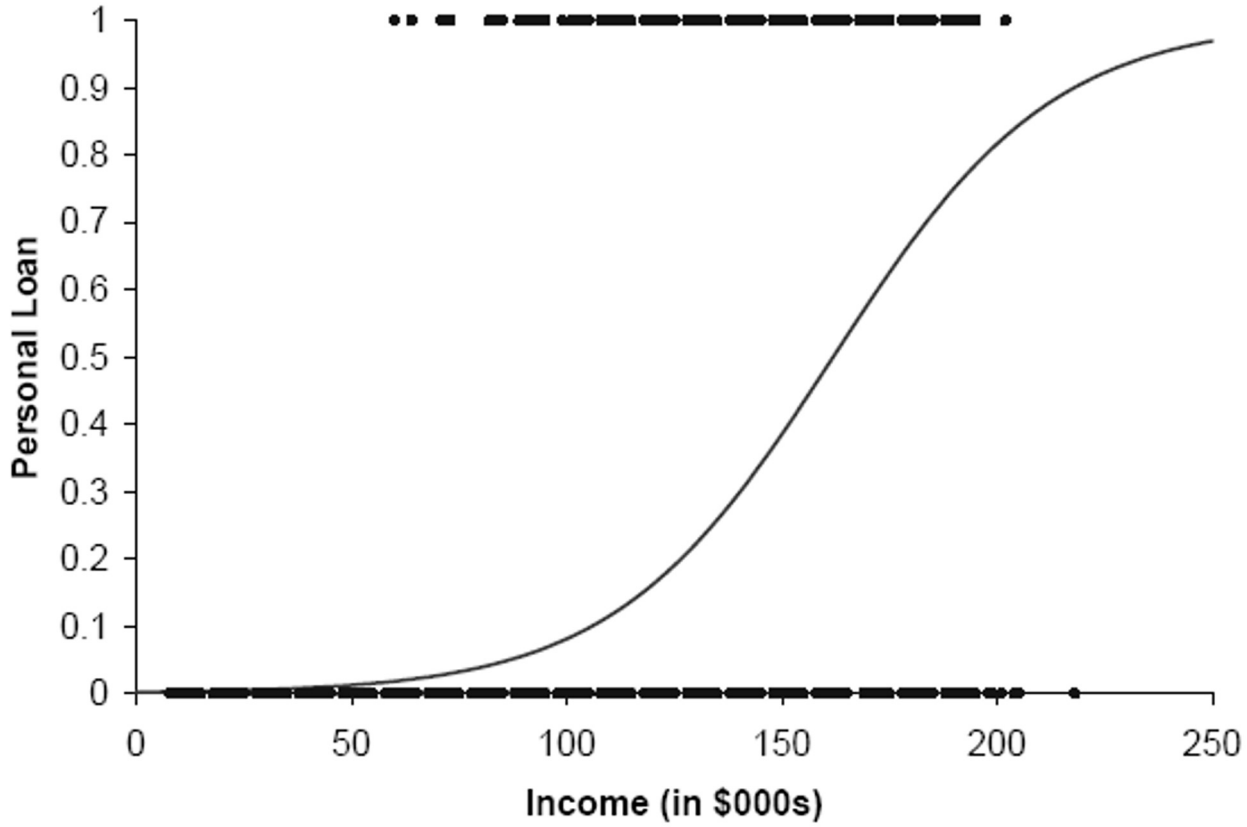
$$\text{Prob}(Personal\ Loan = Yes \mid Income = x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

Fitted coefficients (more later): $b_0 = -6.3525$, $b_1 = -0.0392$

$$P(Personal\ Loan = Yes \mid Income = x) = \frac{1}{1 + e^{6.3525 - 0.0392x}}$$

Seeing the Relationship

$$P(Personal\ Loan = Yes \mid Income = x) = \frac{1}{1 + e^{6.3525 - 0.0392x}}$$



Last step - classify

Model produces an estimated probability of being a “1”

- Convert to a classification by establishing cutoff level
- If estimated prob. > cutoff, classify as “1”

Ways to Determine Cutoff

- 0.50 is popular initial choice
- Additional considerations (see Chapter 5)
  - Maximize classification accuracy
  - Maximize sensitivity (subject to min. level of specificity)
  - Minimize false positives (subject to max. false negative rate)
  - Minimize expected cost of misclassification (need to specify costs)

Fitting the Model

```
# run logistic regression
# use glm() (general linear model) with family = "binomial" to fit a logistic
# regression.
logit.reg <- glm(Personal.Loan ~ ., data = train.df, family = "binomial")
options(scipen=999)
summary(logit.reg)
```

Output

```
                                 Estimate  Std. Error z value             Pr(>|z|)
(Intercept)                   -12.6805628   2.2903370  -5.537  0.0000000308 ***
Age                            -0.0369346   0.0848937  -0.435       0.66351
Experience                      0.0490645   0.0844410   0.581       0.56121
Income                          0.0612953   0.0039762  15.416 < 0.0000000000000002 ***
Family                          0.5434657   0.0994936   5.462  0.0000000470 ***
CCAvg                           0.2165942   0.0601900   3.599       0.00032 ***
EducationGraduate               4.2681068   0.3703378  11.525 < 0.0000000000000002 ***
EducationAdvanced/Professional  4.4408154   0.3723360  11.927 < 0.0000000000000002 ***
Mortgage                        0.0015499   0.0007926   1.955       0.05052 .
Securities.Account             -1.1457476   0.3955796  -2.896       0.00377 **
CD.Account                      4.5855656   0.4777696   9.598 < 0.0000000000000002 ***
Online                         -0.8588074   0.2191217  -3.919  0.0000888005 ***
CreditCard                     -1.2514213   0.2944767  -4.250  0.0000214111 ***
---
```

coefficients for logit

β is the factor by which the odds of belonging to class 1 increase

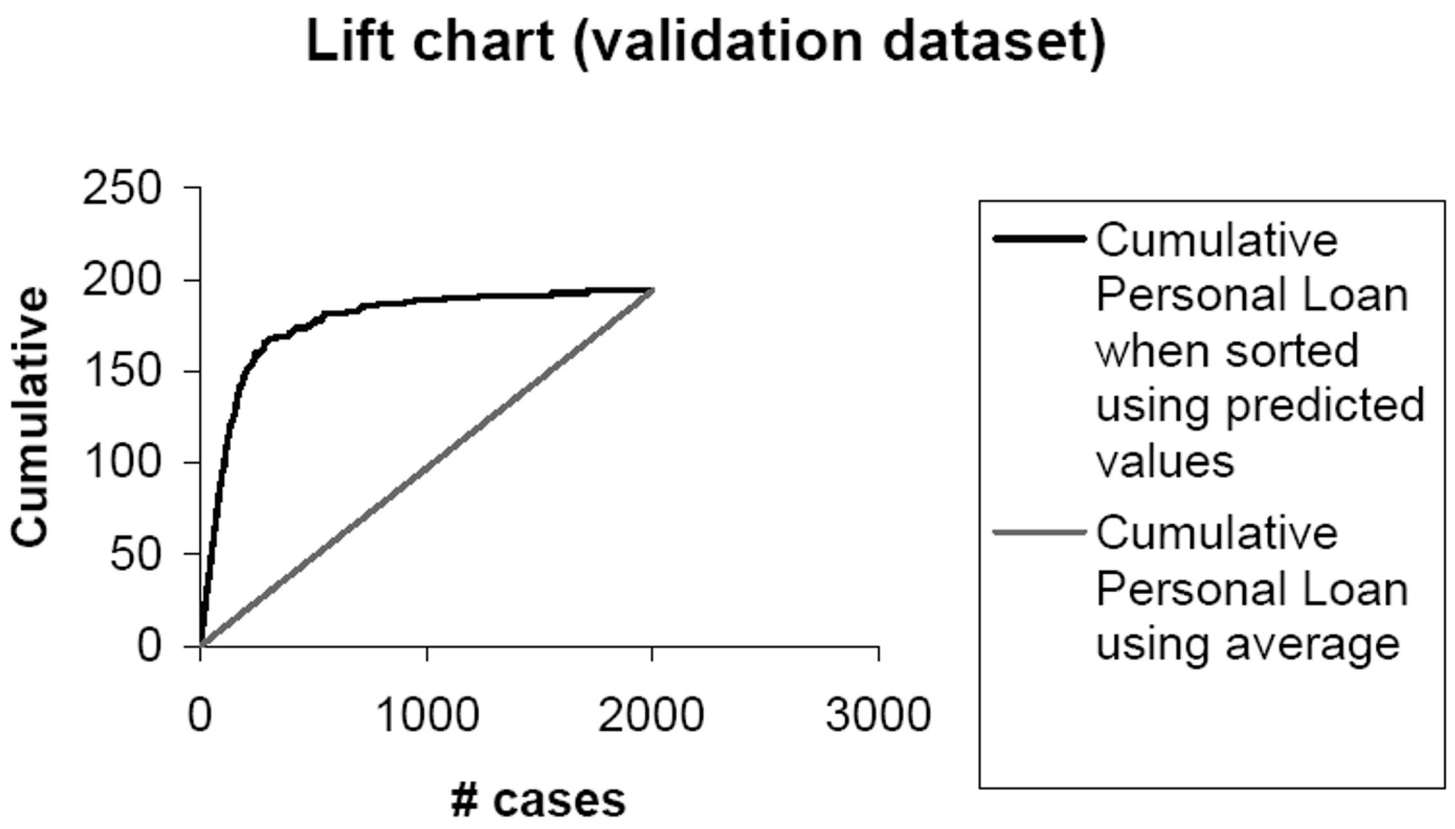
Converting to Probability

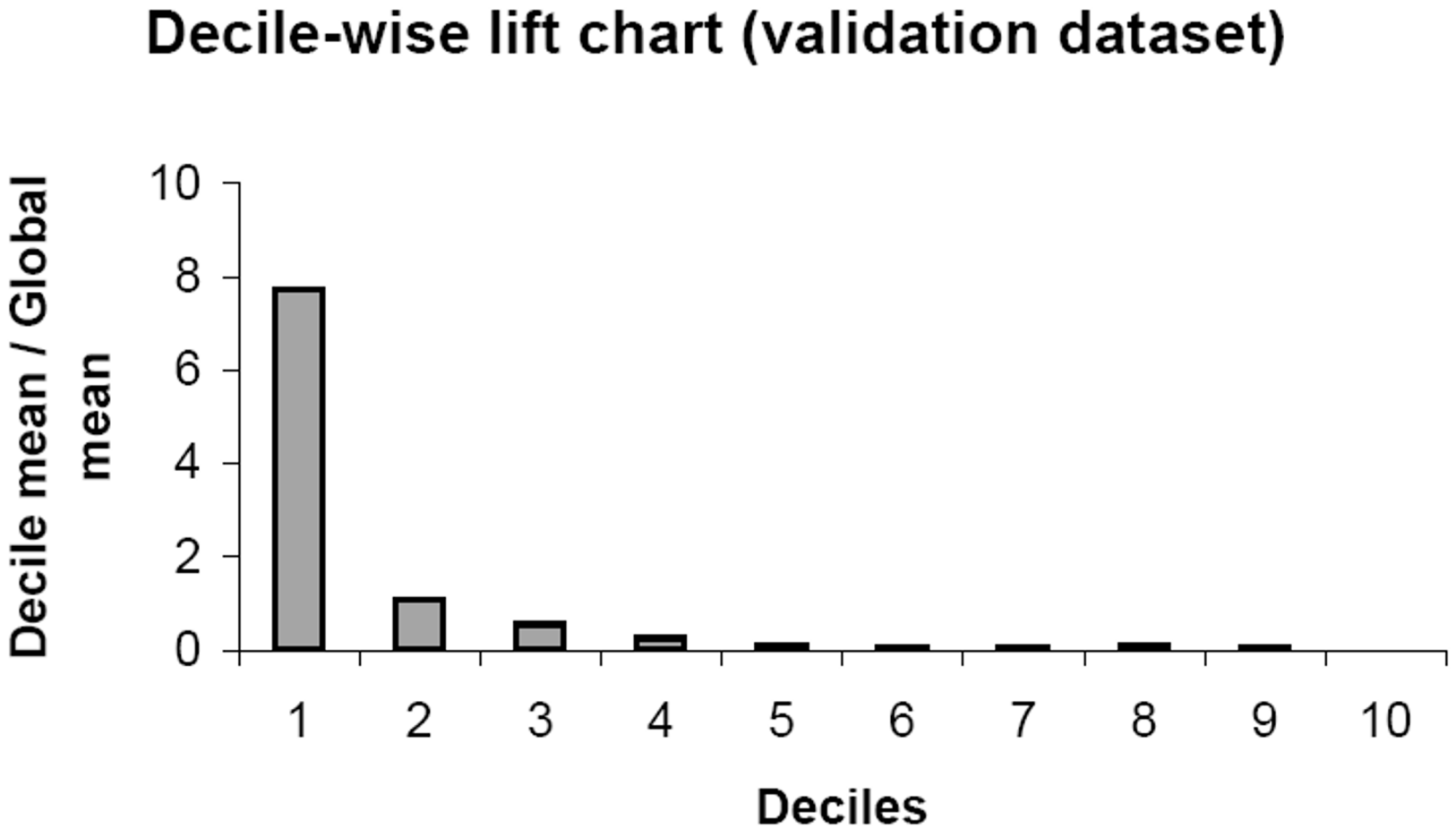
$$p = \frac{Odds}{1 + Odds}$$

Loan Example:
Evaluating Classification Performance

Performance measures: Confusion matrix and % of misclassifications

More useful in this example: **lift**




Decile-wise lift chart (validation dataset)
Decile mean / Global mean
10
8
6
4
2
0
1
2
3
4
5
6
7
8
9
10
Deciles

Multicollinearity

Problem: As in linear regression, if one predictor is a linear combination of other predictor(s), model estimation will fail

- Note that in such a case, we have at least one redundant predictor

Solution: Remove extreme redundancies (by dropping predictors via variable selection, or by data reduction methods such as PCA)

Variable Selection

- This is the same issue as in linear regression
- The number of correlated predictors can grow when we create derived variables such as **interaction terms** (e.g. *Income x Family)*, to capture more complex relationships
- Problem: Overly complex models have the danger of overfitting
- Solution: Reduce variables via automated selection of variable subsets (as with linear regression)

P-values for Predictors

- Test null hypothesis that coefficient = 0
- Useful for review to determine whether to include variable in model
- Important in profiling tasks, but less important in predictive classification

Summary

- Logistic regression is similar to linear regression, except that it is used with a categorical response
- It can be used for explanatory tasks (=profiling) or predictive tasks (=classification)
- The predictors are related to the response Y via a nonlinear function called the *logit*
- As in linear regression, reducing predictors can be done via variable selection
- Logistic regression can be generalized to more than two classes