Master of Technology in Knowledge Engineering

Predictive Analytics and Data Mining

Dr. Zhu Fangming
Institute of Systems Science,
National University of Singapore.
E-mail: isszfm@nus.edu.sg

Objectives

- To introduce predictive analytics and predictive modelling
- To discuss major predictive modelling issues and considerations

Agenda

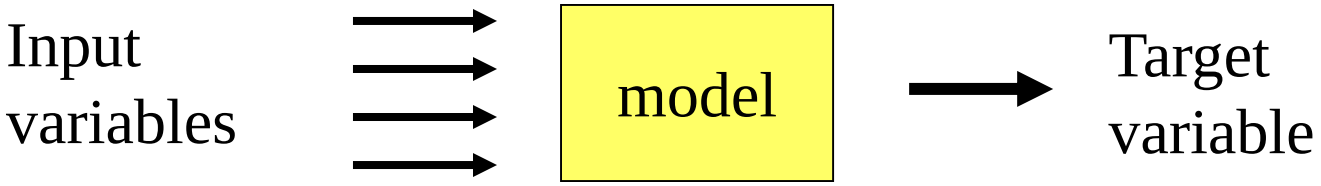
- Predictive modelling concepts and examples
- Predictive modelling process and issues

Predictive Analytics

- Descriptive vs Predictive data mining
- Unsupervised learning vs supervised learning
- Predictive analytics is to build a model for prediction. The model builders will extract knowledge from historical data and represent it such a form that we can apply the resulting model to new situations.
- Typical predictive modelling tasks: classification models and regression models

Predictive Modeling

- For prediction:



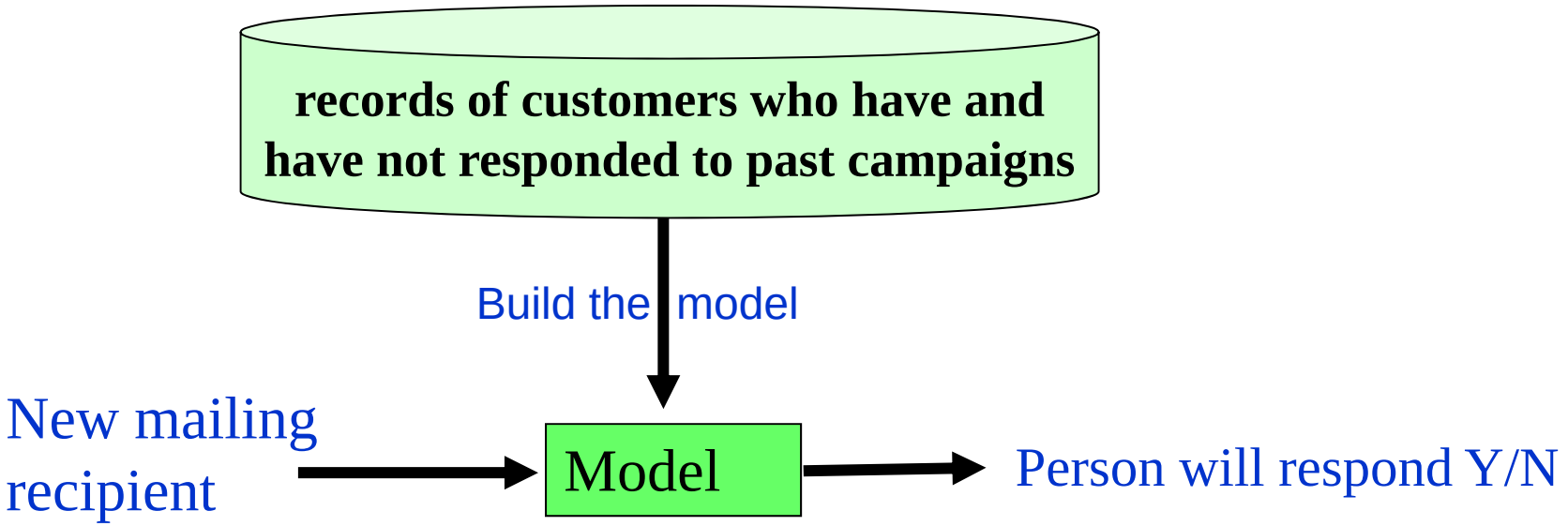
- Key points
  - The model is built automatically - no user intervention is required
  - The user must test & validate the model before use
  - Once built, the model can be applied to fresh data

Categories of Prediction Model

- Event Prediction - will something happen?
  - E.g. will customer respond to a campaign?
  - E.g. will customer buy a product?
- Value Prediction - predicting the value of something
  - E.g. how much money will a person spend?

Event Prediction

- Predicting a T/F event, e.g. persons response to a mailing



Purchase Propensity Example

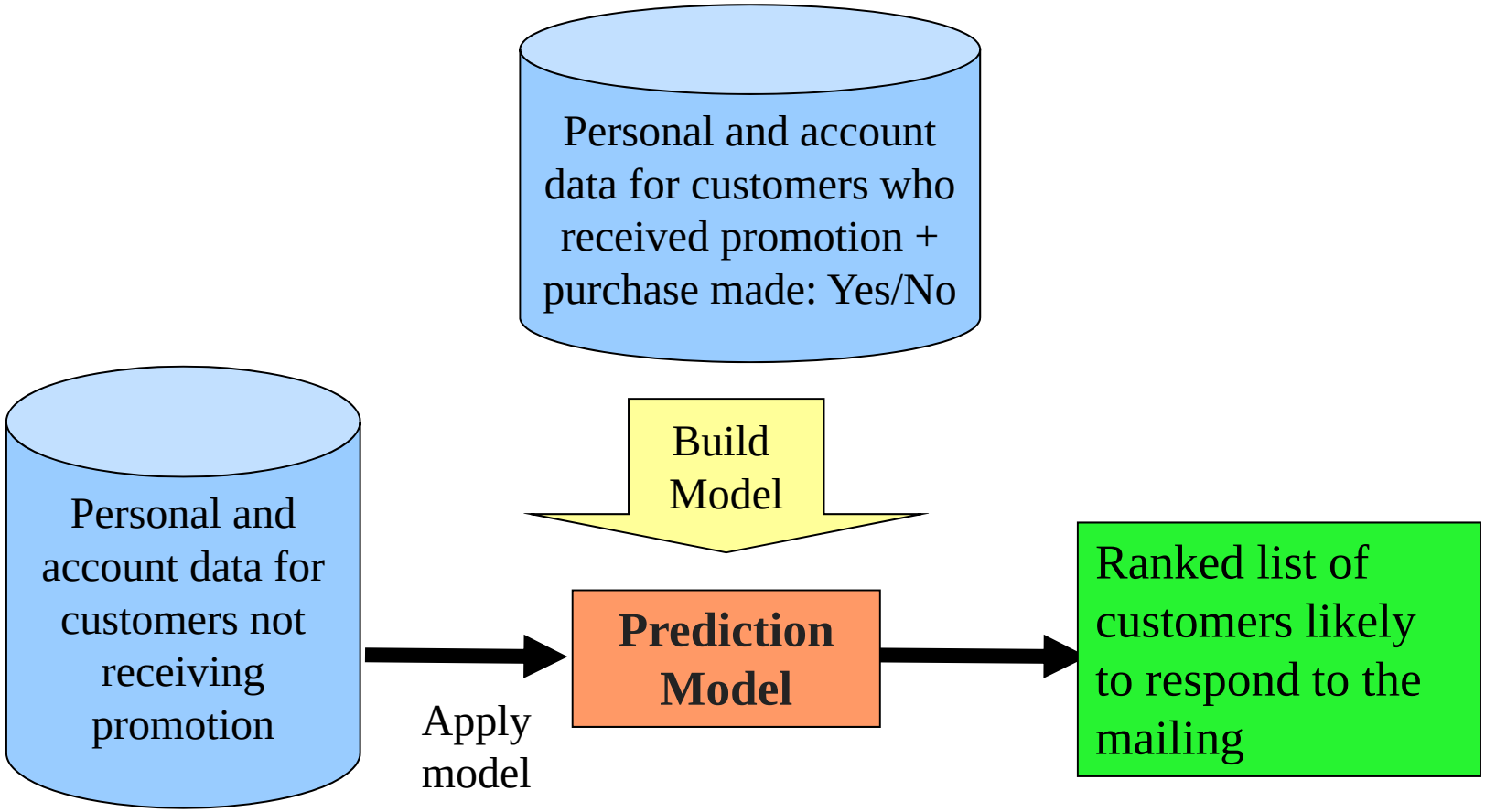
- A bank sends occasional mailings to its customers advertising an investment product called PEP (Personal Equity Plan)
- The bank wishes to initiate a fresh mailing to customers who have not previously been mailed - due to the cost the bank would like to mail only customers likely to respond and buy a PEP
- Goal = build a model to predict which customers are likely to purchase a PEP, apply this to the overall customer base to obtain a mailing list

Purchase Propensity Example

The marketing department keeps customer records, including demographic and account information:

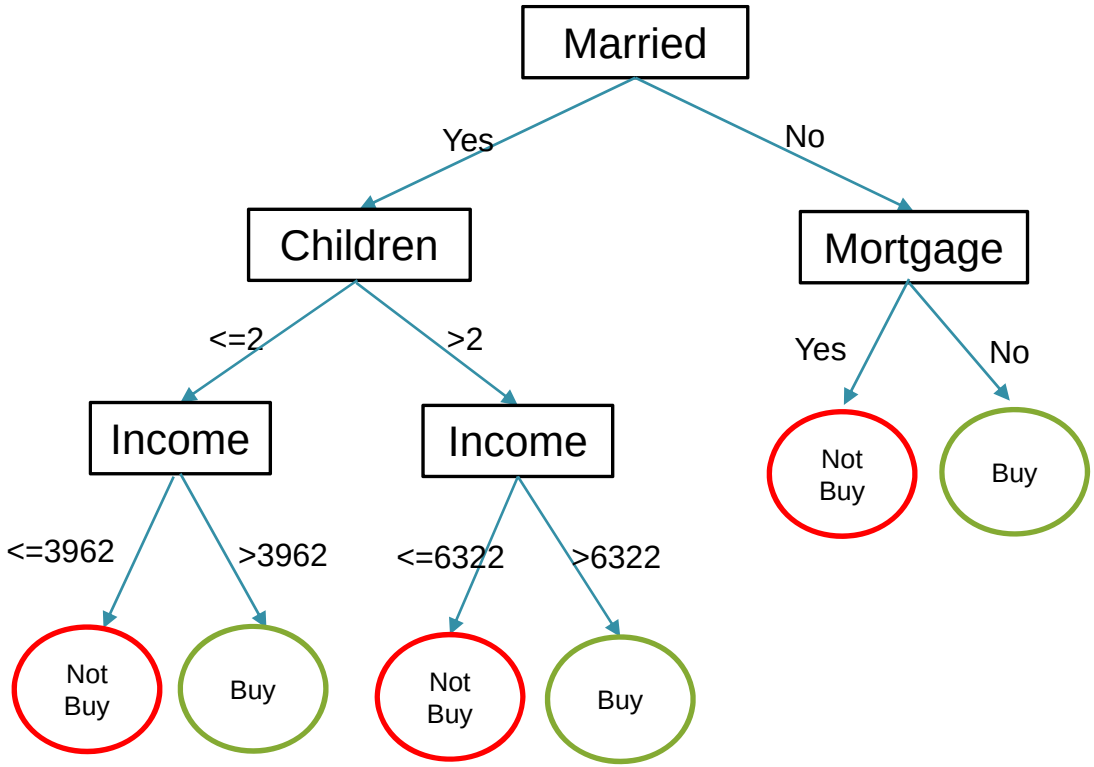
id	a unique identification number
age	age of customer in years
sex	M, F
region	inner-city, rural, suburban, town
income	customer income
married	Y, N
children	number of children
car	does the customer own a car?
save_acct	does the customer have a savings account?
curr_acct	does the customer have a current account?
mortgage	does the customer have a mortgage?
pep	did the customer buy a PEP after the last mailshot?

Purchase Propensity Example



Purchase Propensity Example

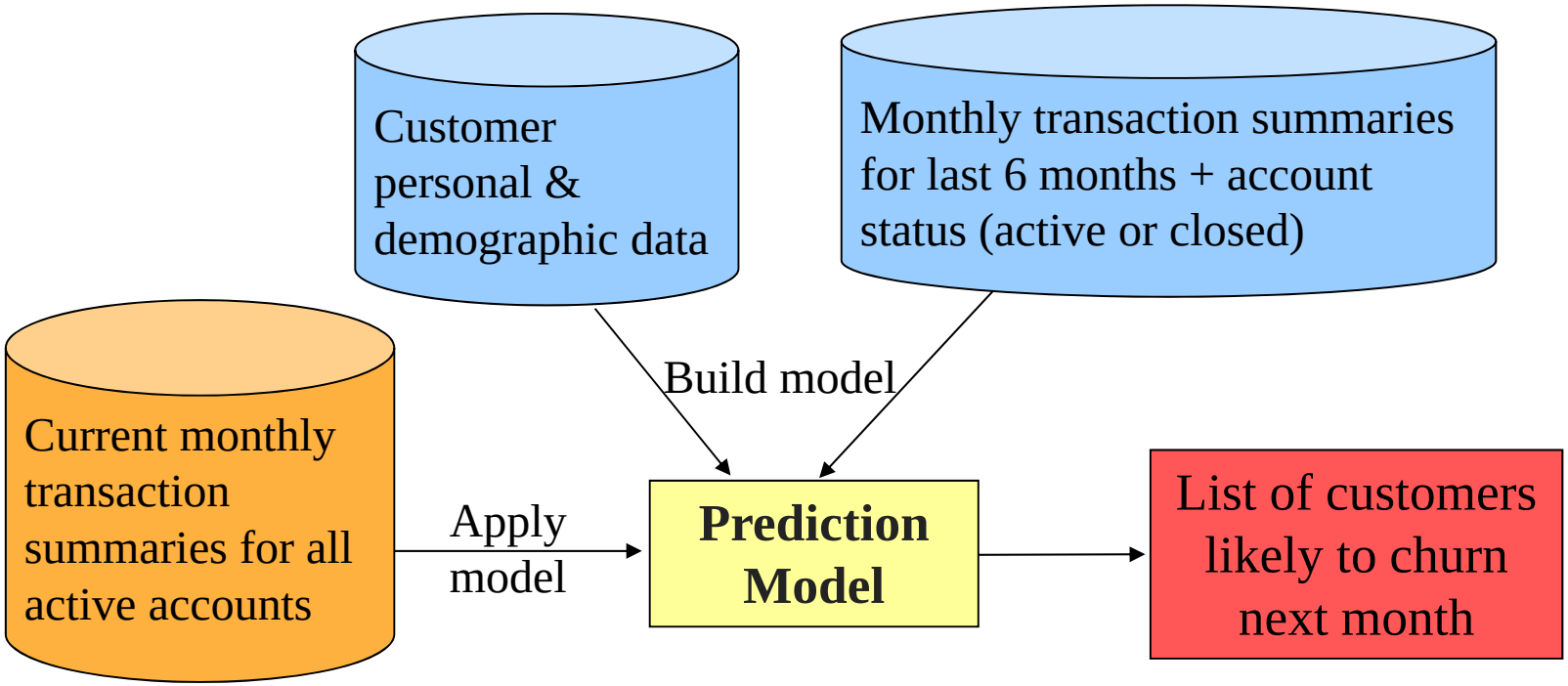
One of the decision tree models built from the data



Churn Modeling Example

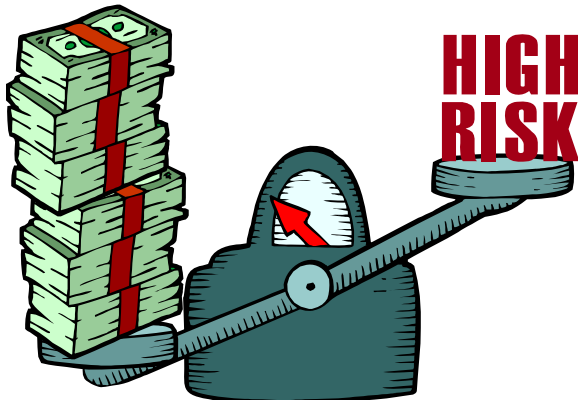
- A Telco wishes to improve customer satisfaction and reduce the rate of churn in its mobile phone account holders
- Typical Data Available
  - Customer account information
    - payment plan, handset type, date of signup, additional services used, account status (active or closed) ….
  - Records of all calls made over the last 6 months
  - Customer personal information
  - Customer satisfaction survey results
- Task: Given customer information for the past N months, predict who is likely to attrite next month.

Churn Modeling Example



Credit Risk Modeling

- Situation: Person applies for a loan
- Task: Should a bank approve the loan?
- Note: People who have the best credit don't need the loans, and people with worst credit are not likely to repay. Bank's best customers are in the middle

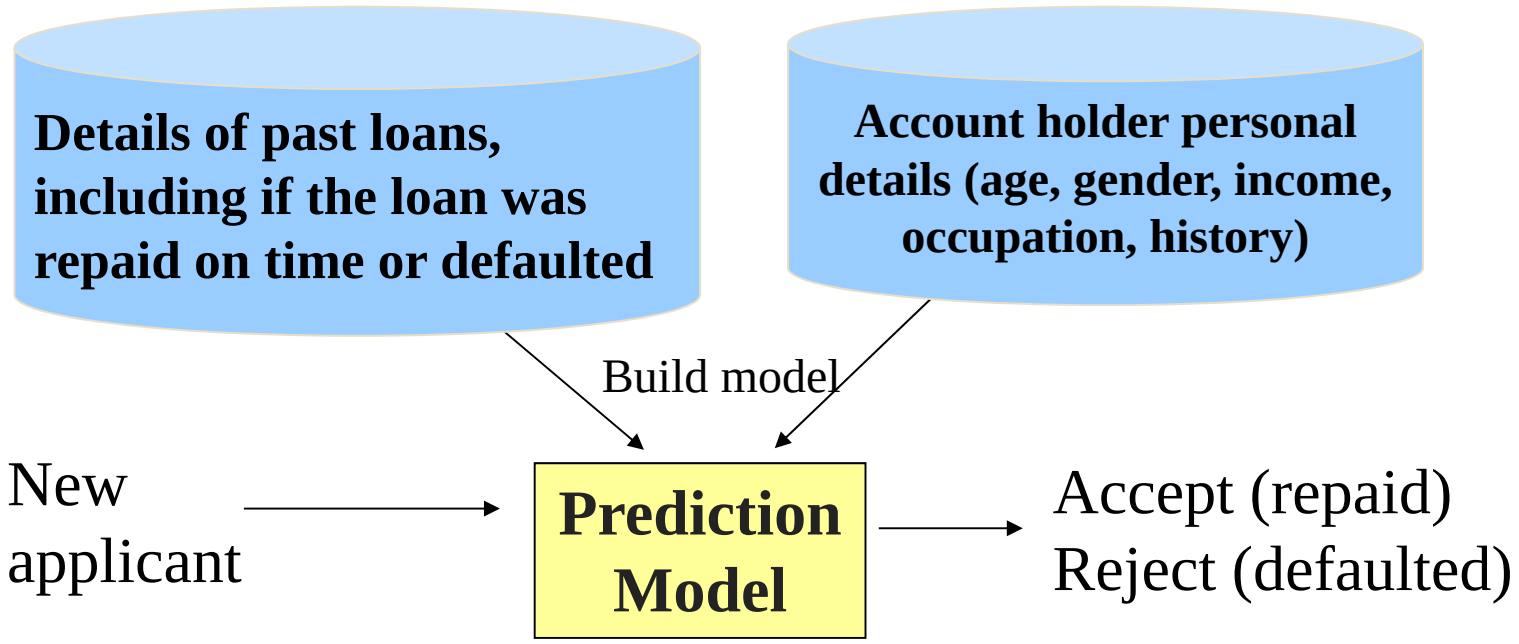


Credit Risk Modeling

- Banks commonly make credit decisions based on simple credit scoring
  - Candidate is asked a series of questions, a score is assigned to each of their answers (e.g. 0 →10)
  - Scores are added and compared against a simple threshold
- Widely deployed in many countries but can be inaccurate – reject good candidates, accept poor ones

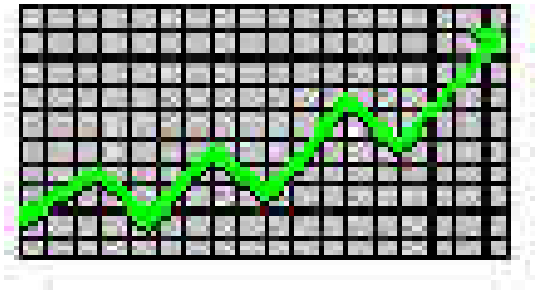
Credit Risk Modeling

- Banks can mine past records of loans to learn who not to offer future loans



Value Prediction

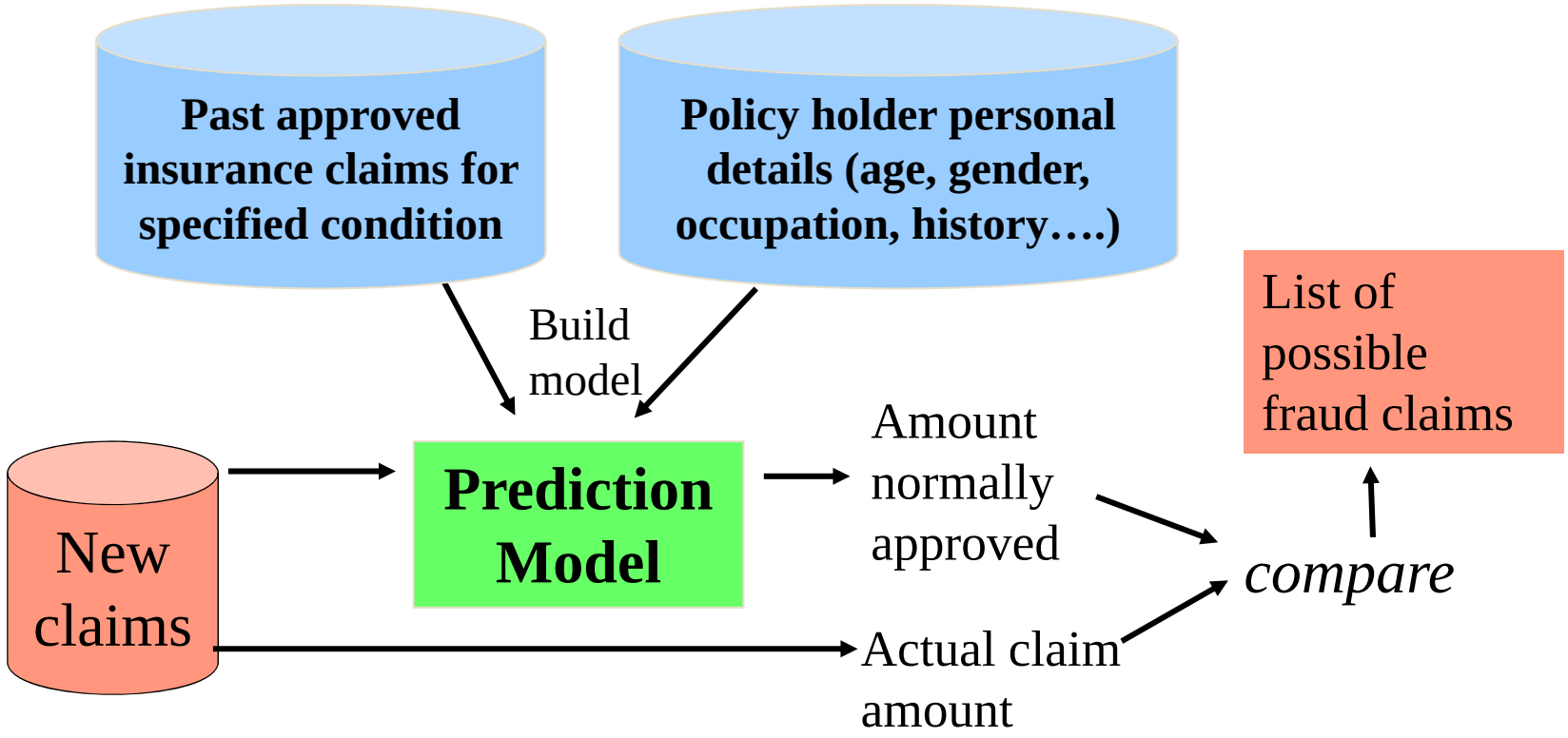
- The Model is used to predict a numerical value, e.g.
  - sales volume
  - stock price
  - $ amount a person will spend
  - cholesterol level
- Common methods
  - Neural networks
  - Linear Regression



Case Study: Insurance Claim Fraud

- A large insurance company has seen a big increase in the number and size of claims against medical policies over the last year – it needs a better mechanism for detecting inflated claims
- Typical data available
  - Medical claims made over the past 3 years
  - Demographic & personal details of its insurance policy holders

Case Study: Insurance Claim Fraud



Core Model Building Steps

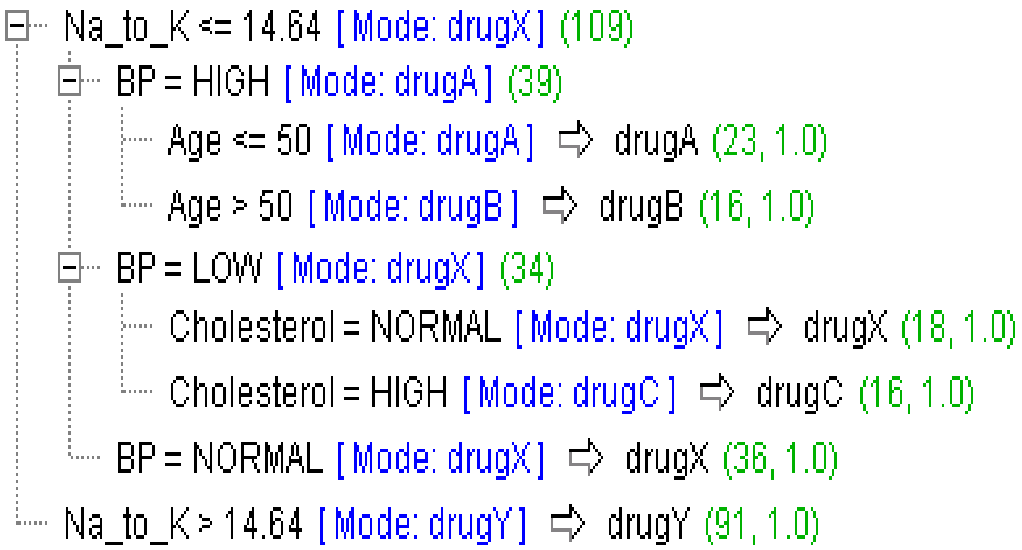
1. Select the modeling method
2. Select model inputs and target
3. Adjust parameters (advanced users)
4. Select training & test set
5. Generate the model
6. Test the model
7. Repeat from 1, 2, 3 or 4 if model is poor

Selecting the Modeling Method

- Decision tree (C5, C4.5, CART)
- Rule sets (C5, C4.5)
- Linear regression
- Logistic regression
- Neural network (MLFF, Radial Basis, Kohonen…)
- Memory-based reasoning (K-NN)
- Support Vector Machine
- Bayes Classification Methods
- Etc….

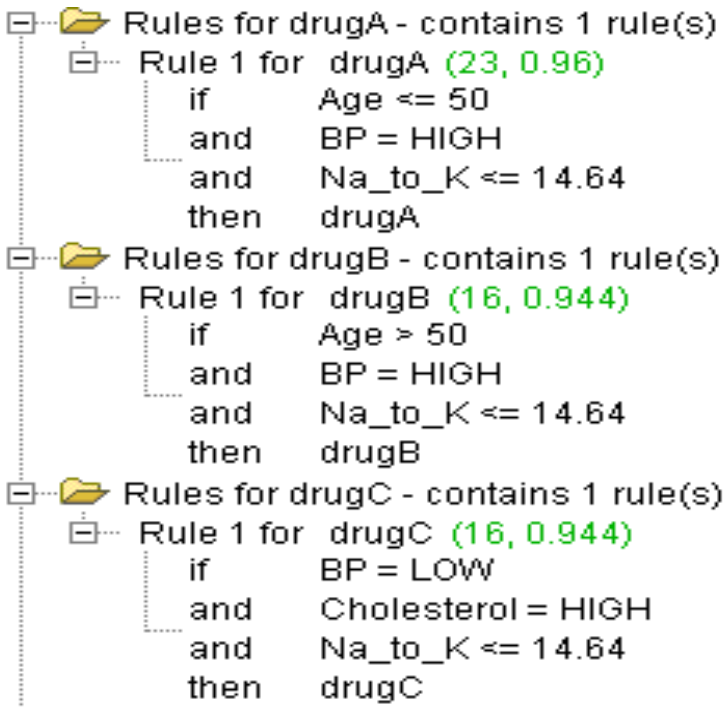
Decision Tree

- Used to predict a discrete target (classification) – not a number



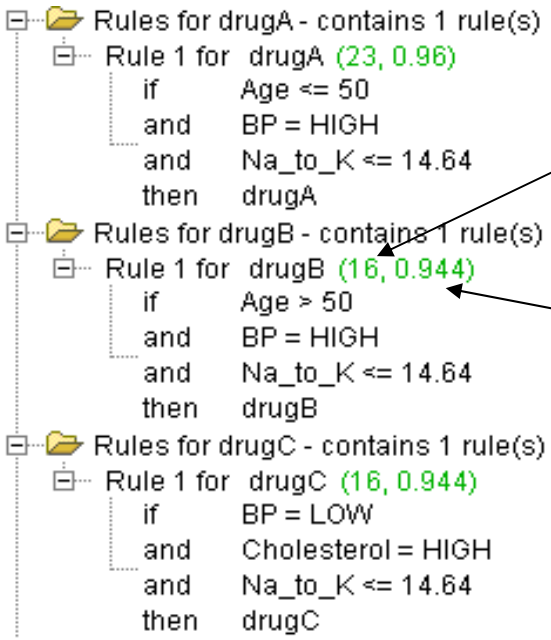
Rule Sets

- Normally displayed in an “if-then” form



Coverage & Accuracy

- Rules and Trees are displayed with coverage and accuracy information



Coverage -- the number of instances to which the rule applies

Accuracy (Confidence) -- the proportion of those instances for which the rule is true

Overtraining / Overfitting

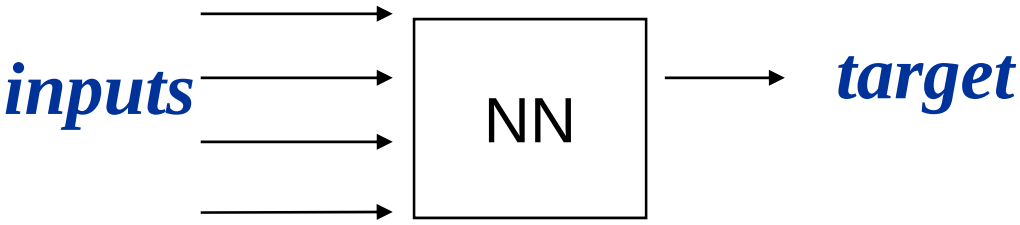
- The unrestrained tree algorithm builds trees that correctly classify every training record
- Problems
  - They generalise poorly!
  - They memorise the training data (including the noise)
- This is called overfitting
- Similar situation occurs with rulesets

Tree Pruning

- Prevents overfitting
- Done by default in most tools
- Stops tree growth when
  - It gets too specialised
    (too few examples covered)
  - It gets too big
    (too many tests)
  - Its error rate stops decreasing
    (when tested on a test set)
- Degree of pruning is user adjustable

Neural Networks

- Make no assumptions about the data
- Can be very accurate
- Handle both numeric targets and categorical targets
- No need to understand NN theory to use, treat as a black box….

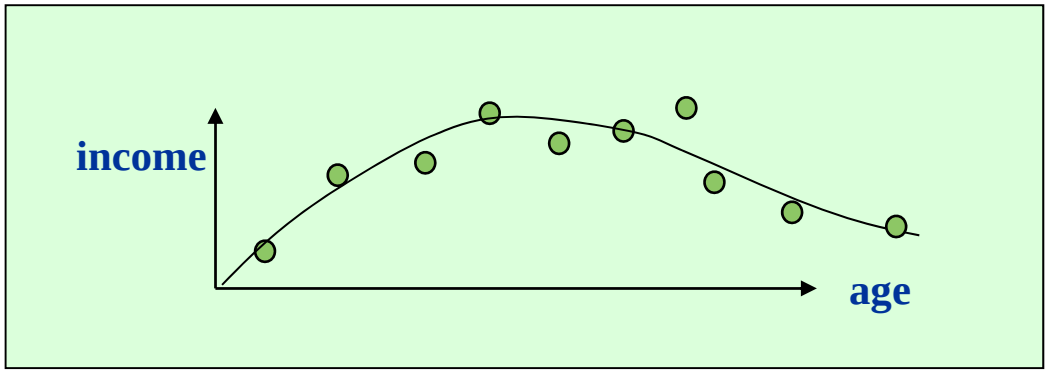


Training Neural Networks

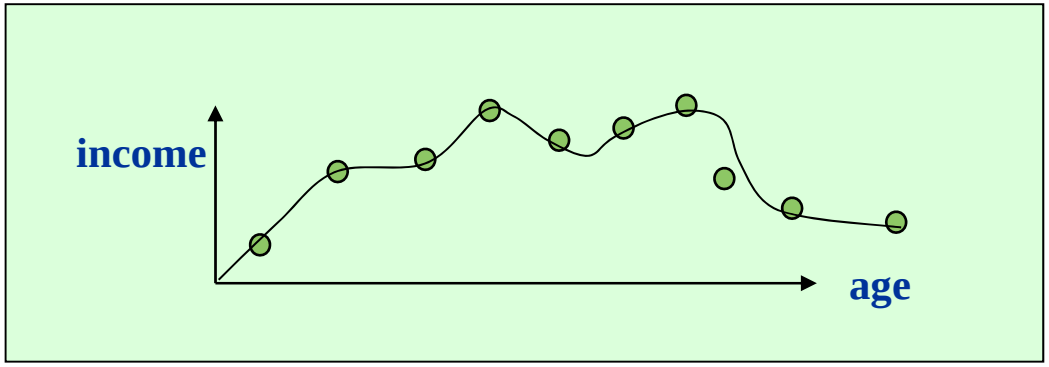
- Require lots of training data
- Can be very slow!
- Limit training by
  - the time taken
  - number or training iterations
  - the accuracy
- Random initial weights means different NNs can be generated from the same data

Neural Networks

- Can fit any shape curve if enough training examples are given



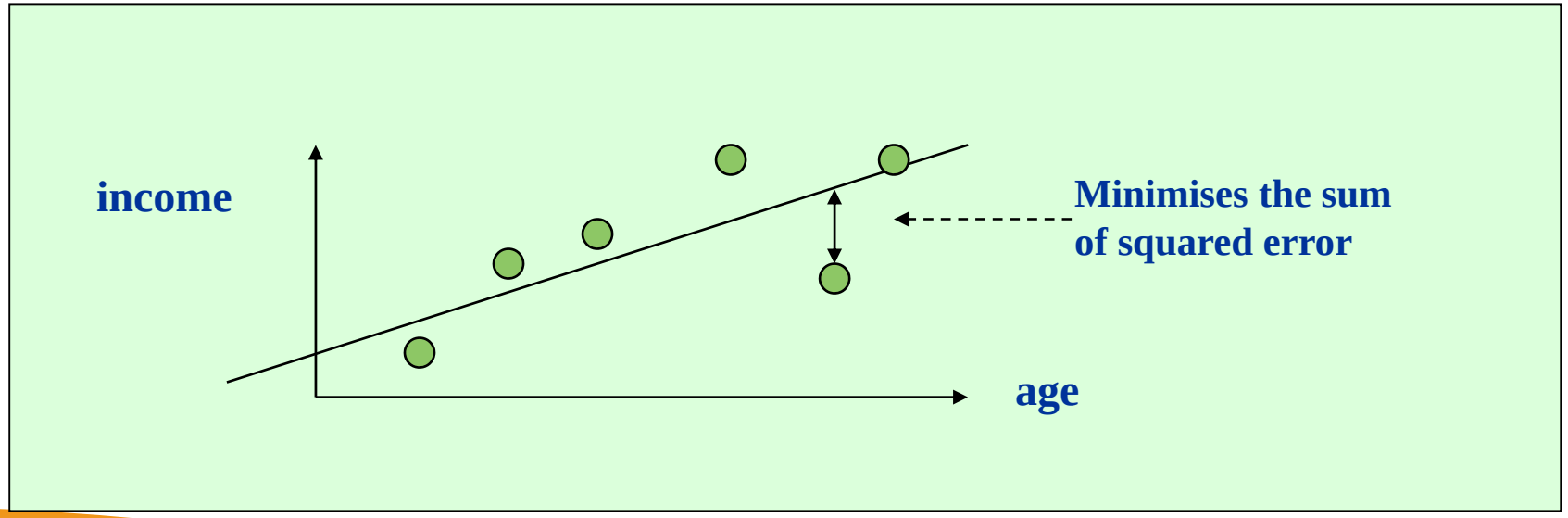
- Beware of overfitting! Can happen if you train the NN for too long or give the NN too many internal nodes and/or layers



Regression Methods

Linear Regression

- Use for numeric targets
- Good if you know the target changes linearly (as a straight line)
- Assumes the model: t = ax +by +cz + d etc.



Regression Methods

- ***Logistic Regression*:** alternative to linear regression
  - Designed for classification problems
  - Tries to estimate class probabilities directly
  - Uses this linear model:

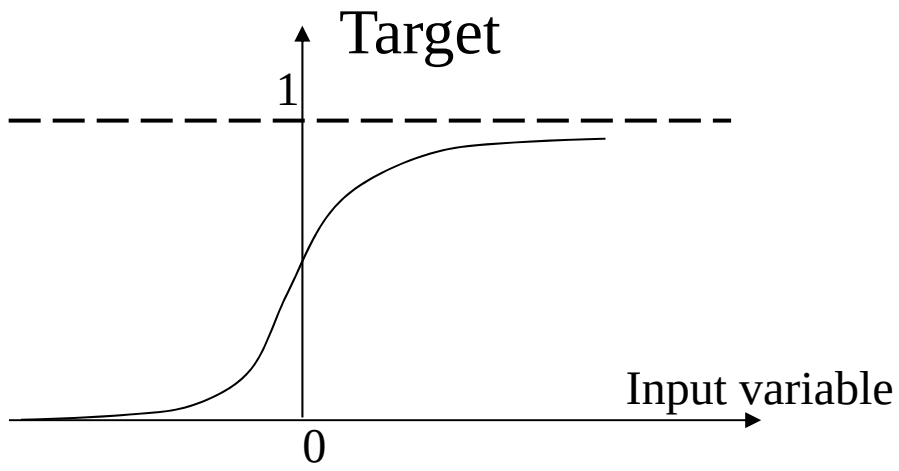
$$\ln\left(\frac{P}{1-P}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_i x_i$$

P= Class probability

Regression Methods

Logistic Regression

For one input variable we can draw the logistic function as

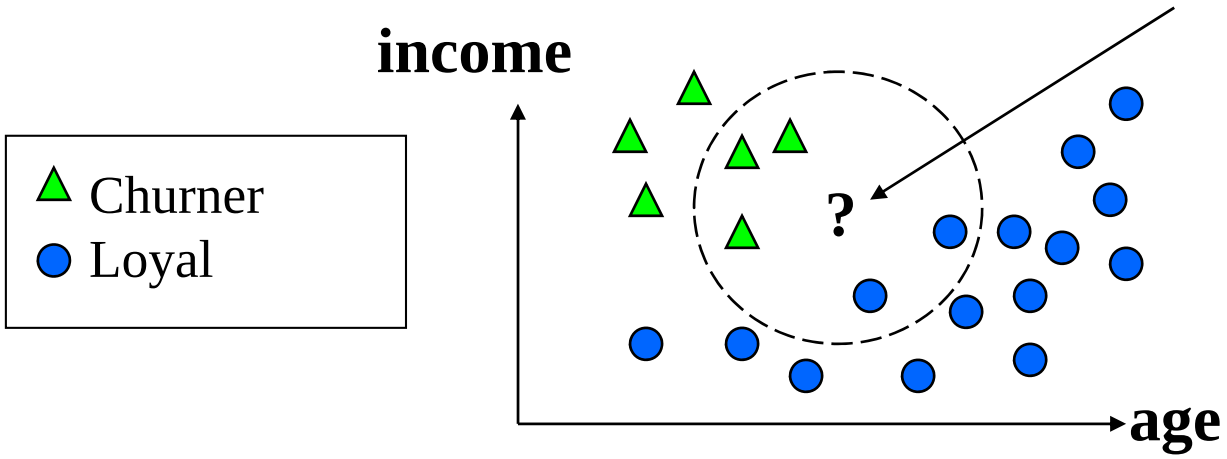


Which is a good match for many T/F prediction situations

The transformation ln(p/1-p) turns this into a straight line and hence suitable for linear regression (p = prob(target))

The K-Nearest Neighbour Algorithm

- Find the K nearest training records to the unknown case
- For classification, use the target value of the majority
- For predicting numbers, take the weighted average of all K target values



If K=5 we classify the unknown person as a potential Churner

Detecting Churners (highly simplified situation)

Selecting the Model Inputs

- Reducing dimensionality helps the modeling algorithms
  - Select variables that are related to the target
  - Eliminate variables that are redundant (e.g. duplicates)
  - E.g. reduce 200 variables down to 50 or less
- Use domain knowledge
  - Eliminate known irrelevant and duplicate variables
  - Keep known relevant variables
- Use visualization and other exploration techniques to investigate the “maybe” variables
- Try to be selective but if uncertain, include a variable

Selecting the Model Inputs

- **Simplification and Consolidation**
- What if 4 separate fields exist as below?
  - Single = 0,1
  - Married = 0,1
  - Divorced = 0,1
  - Widowed = 0,1
- For visualisation and rule/tree models, best combine into one field called “marital status”
  - Values = single, married, divorced, widowed
- For NN and regression models, best to leave as they are (see next slide)

Selecting the Model Inputs

- NN and regression methods do math calculations!
- How do they handle:
  - Marital status = single, married, divorced, widowed?
- Either convert to
  - Marital status = 0,1,2,3 (use if an ordering is implicit)
- Or create 4 new T/F variables
  - Single = 0,1
  - Married = 0,1
  - Divorced = 0,1
  - Widowed = 0,1

} use if there is no ordering

Selecting the Model Inputs

- Do not select discrete fields with large numbers of categories
  - E.g. account number, IC number, address, postalcode, etc.
- Perform some transformations
  - E.g. age = current date - date of birth
  - E.g. transform states into groups, e.g. western, eastern

Selecting Training & Test sets

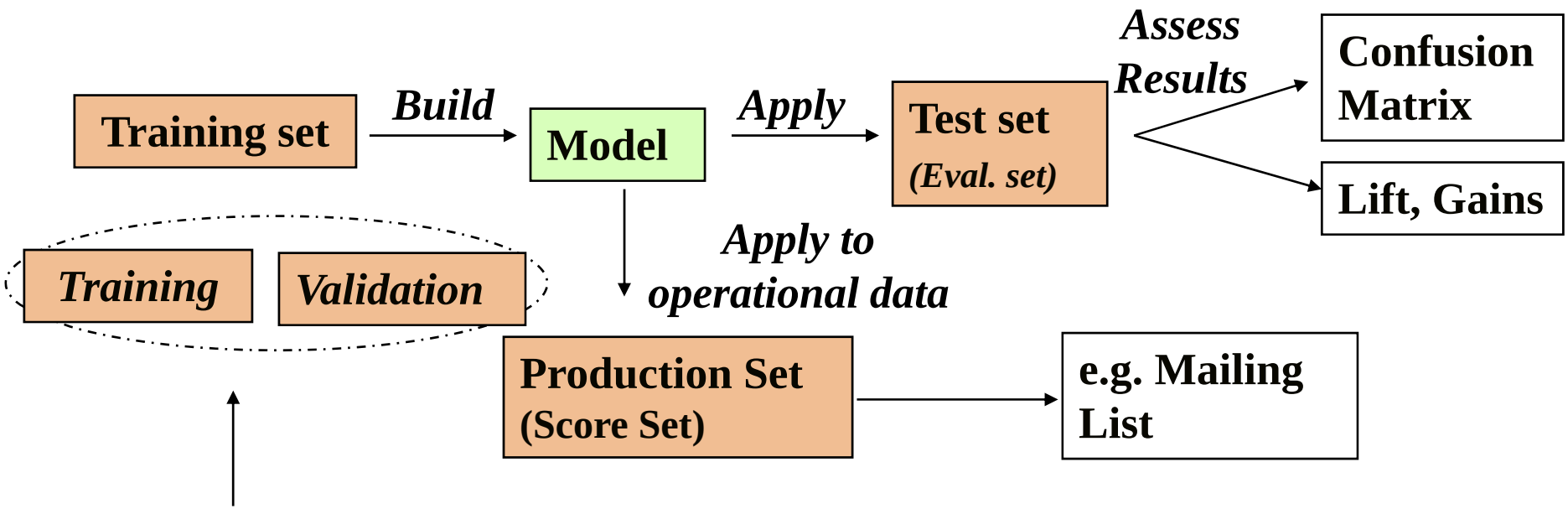
- In order to test the model, it is necessary to divide the data into training and test sets, e.g.

- Select the training data randomly.
- Usually select the test set to be everything not in the training set

Selecting Training & Test Sets

- Partition: Training-and-Testing
  - use two independent data sets, e.g., training set (2/3), test set (1/3)
  - used for data set with large number of samples
- Cross-validation
  - divide the data set into *k* subsamples
  - use *k-1* subsamples as training data and one sub-sample as test data — *k*-fold cross-validation
  - for data set with moderate size

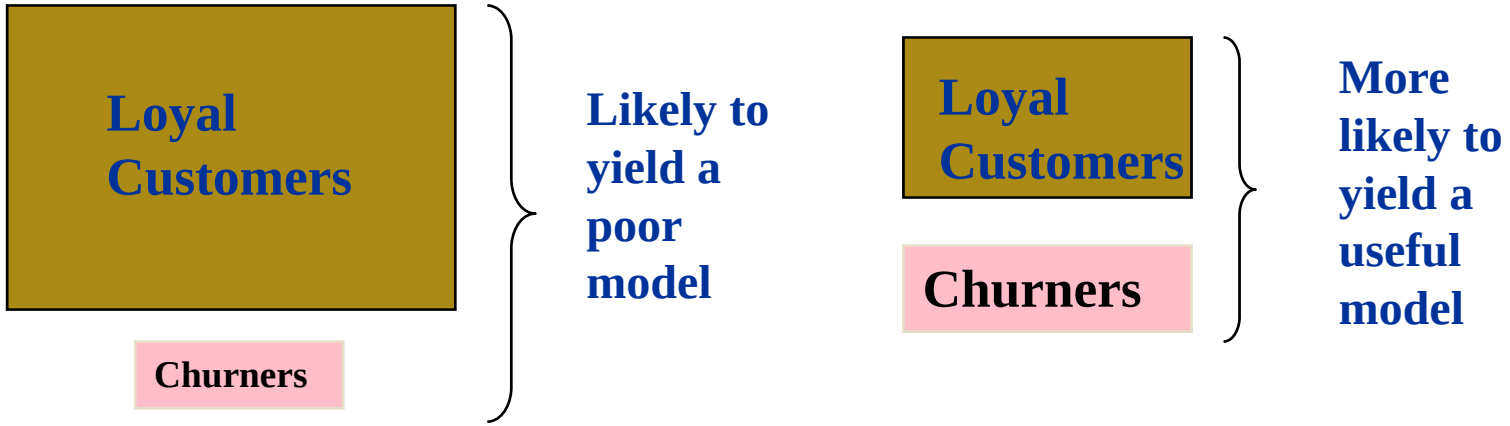
Data Sets Used in Model Building



Sometimes the training set is split into two so that optimisation can be performed during model generation (e.g. rule pruning). In this case the training set is divided into a smaller training set + a validation set.

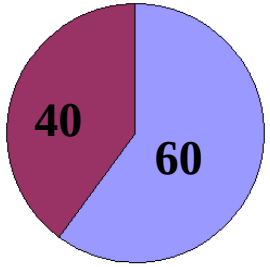
Data Balancing

- Unbalanced training data can result in poor models
- E.g. 9900 records where decision = loyal
  100 records where decision = churner
  - The mined model is likely to be decision = loyal (99% correct!), very accurate but useless for predicting churn!
- Solution = balance the data as much as practical



Data Balancing

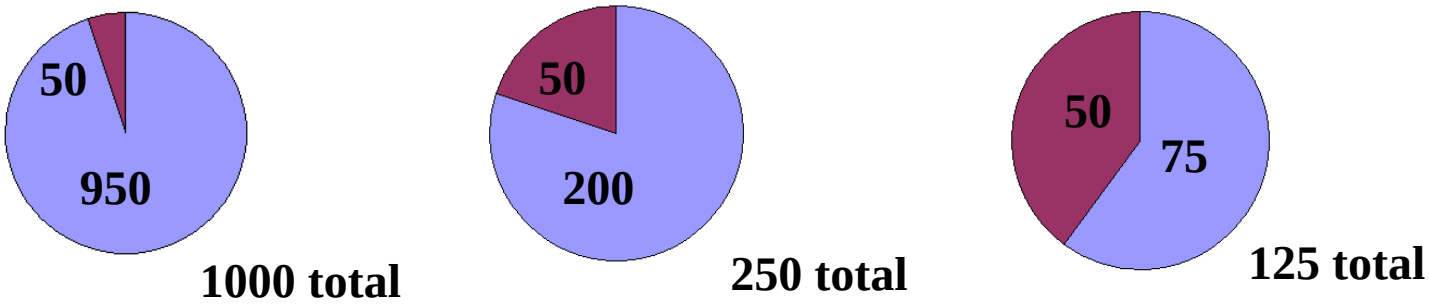
- It's not necessary to achieve equal size categories
  - Between 20/80 and 40/60 is OK



- Common methods use "oversampling"
  - Delete records from the most frequent category
  - Duplicate records in the less frequent category

Data Balancing

- Warning: too much data removal may result in too few training records which may lead to worse results



- Duplicating records may give them too much importance and result in overfitting
- NOTE: always test models on the unbalanced data

Model Testing and Evaluation

- The overall test result (e.g. 85% correct decisions) is useful but often too general a measure
- Must look at the confusion matrix

Classification Evaluation Metrics: Confusion Matrix

Confusion Matrix:

Actual class\Predicted class	Predicted C_1	Predicted ¬ C_1
Actual C_1	**True Positives (TP)**	**False Negatives (FN)**
Actual¬ C_1	**False Positives (FP)**	**True Negatives (TN)**

- It may have extra rows/columns to provide totals

Classification Evaluation Metrics

A\P	C	¬C	
C	**TP**	**FN**	**P**
¬C	**FP**	**TN**	**N**
	P'	**N'**	**All**

- Accuracy = (TP + TN)/All
- Error rate: 1 – accuracy, or
  - Error rate = (FP + FN)/All
- Sensitivity: True Positive Rate
  - Sensitivity = TP/P
- Specificity: True Negative Rate
  - Specificity = TN/N

Classification Evaluation Metrics (cont.)

- **Precision**: exactness – what % of patterns that the classifier labeled as positive are actually positive

$$precision = \frac{TP}{TP + FP}$$

- **Recall:** completeness – what % of positive patterns did the classifier label as positive?

$$recall = \frac{TP}{TP + FN}$$

- ***F* measure (F_1 or *F*-score)**: harmonic mean of precision and recall,

$$F = \frac{2 \times precision \times recall}{precision + recall}$$

The Confusion Matrix –MODEL A

- Used to assess the performance of classification models (models that predict categorical targets)

	Predicted buyer	Predicted Non-buyer	Total
Actual Buyer	200 ***True Positives***	100 ***False Negatives***	300
Actual Non-Buyer	800 ***False Positives***	1900 ***True Negatives***	2700
Total	1000	2000	3000

Beware: this model gets 70% correct (2100/3000, overall accuracy) but only 66% of the actual buyers (200/300) are captured by the model.

The Confusion Matrix – Model B

- Test results for a second model
- Is this a better model than the previous one?

	Predicted buyer	Predicted Non-buyer	Total
Actual Buyer	280 ***True Positives***	20 ***False Negatives***	300
Actual Non-Buyer	1000 ***False Positives***	1700 ***True Negatives***	2700
Total	1280	1720	3000

This model gets only 66% correct (1980/3000) but 93.3% of the actual buyers (280/300) are captured by the model.

Classification Evaluation Metrics: Exercise

	Accuracy	Error rate	Sensitivity	Specificity	Precision	Recall	F
MODEL A							
MODEL B							

Which model is better, Model A or Model B?

Predicting Numeric Targets

- Cannot use a confusion matrix to show results since there are no correct/incorrect decisions
- Instead measure error from the known value
- Common measure is SSE, RMS error, etc.

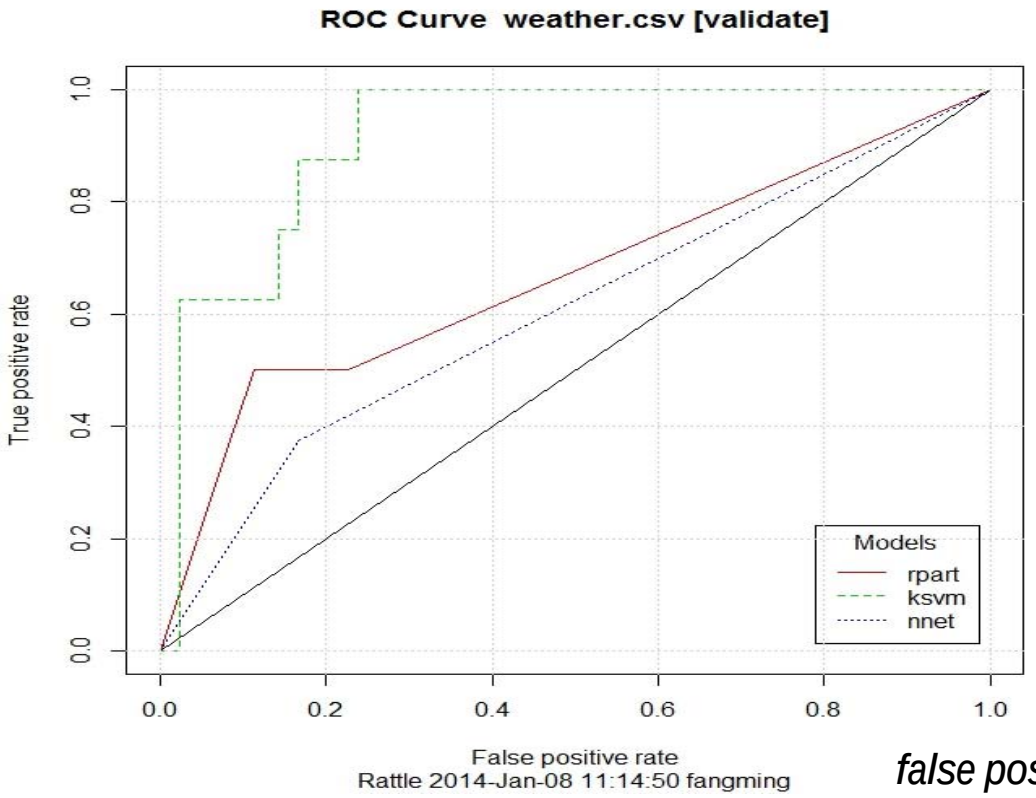
Real value	Predicted value	Error	Squared error
20.5	27.1	6.6	43.56
30	28.5	-1.5	2.25
…	…	…	….
		Total	SSE
		Av. SE	SSE/N
		RMS	root(SSE/(N-P))

Adjusts estimate to take into account complexity of model, where P = number of predictor variables +1 ⟶

Model Assessment with ROC Chart

The ROC chart graphically displays Sensitivity vs. 1-Specificity, or true positive rate vs. false positive rate.

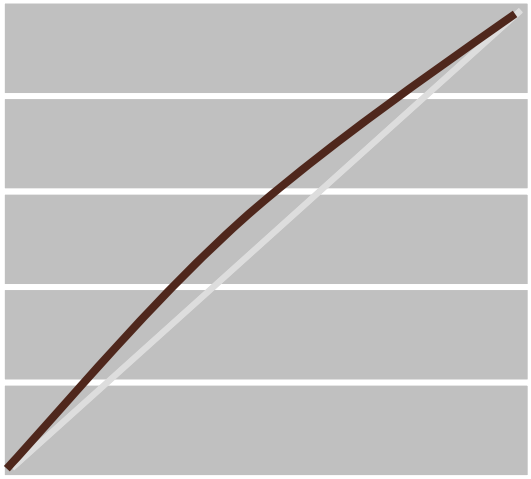
True positive fraction (sensitivity)



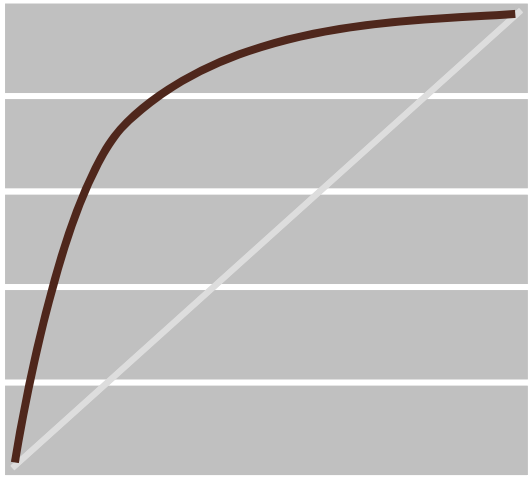
Sensitivity: TP/(TP+FN)
1 – Specificity: FP/(FP+TN)

false positive fraction (1-specificity)

ROC Charts



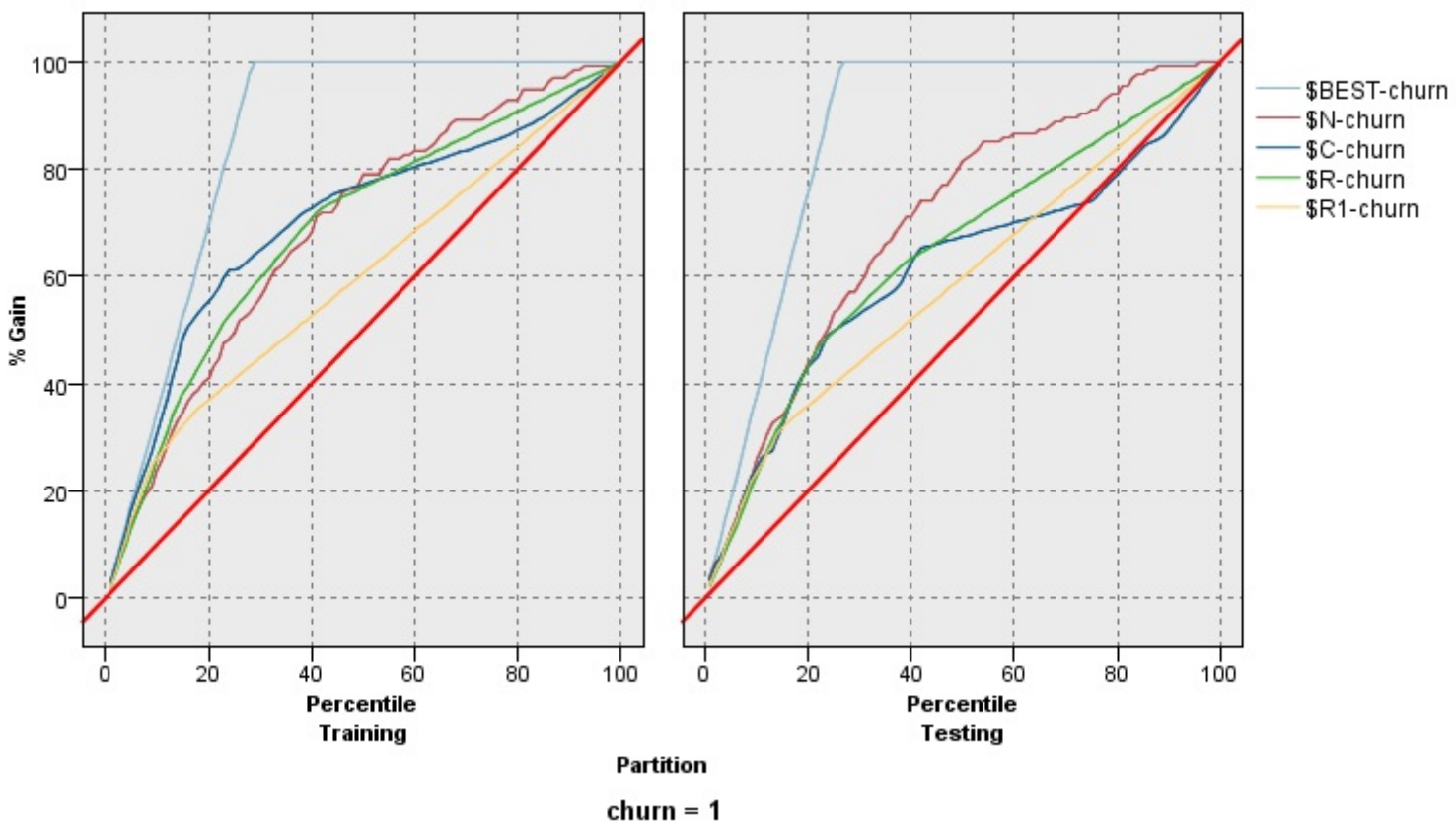
weak model



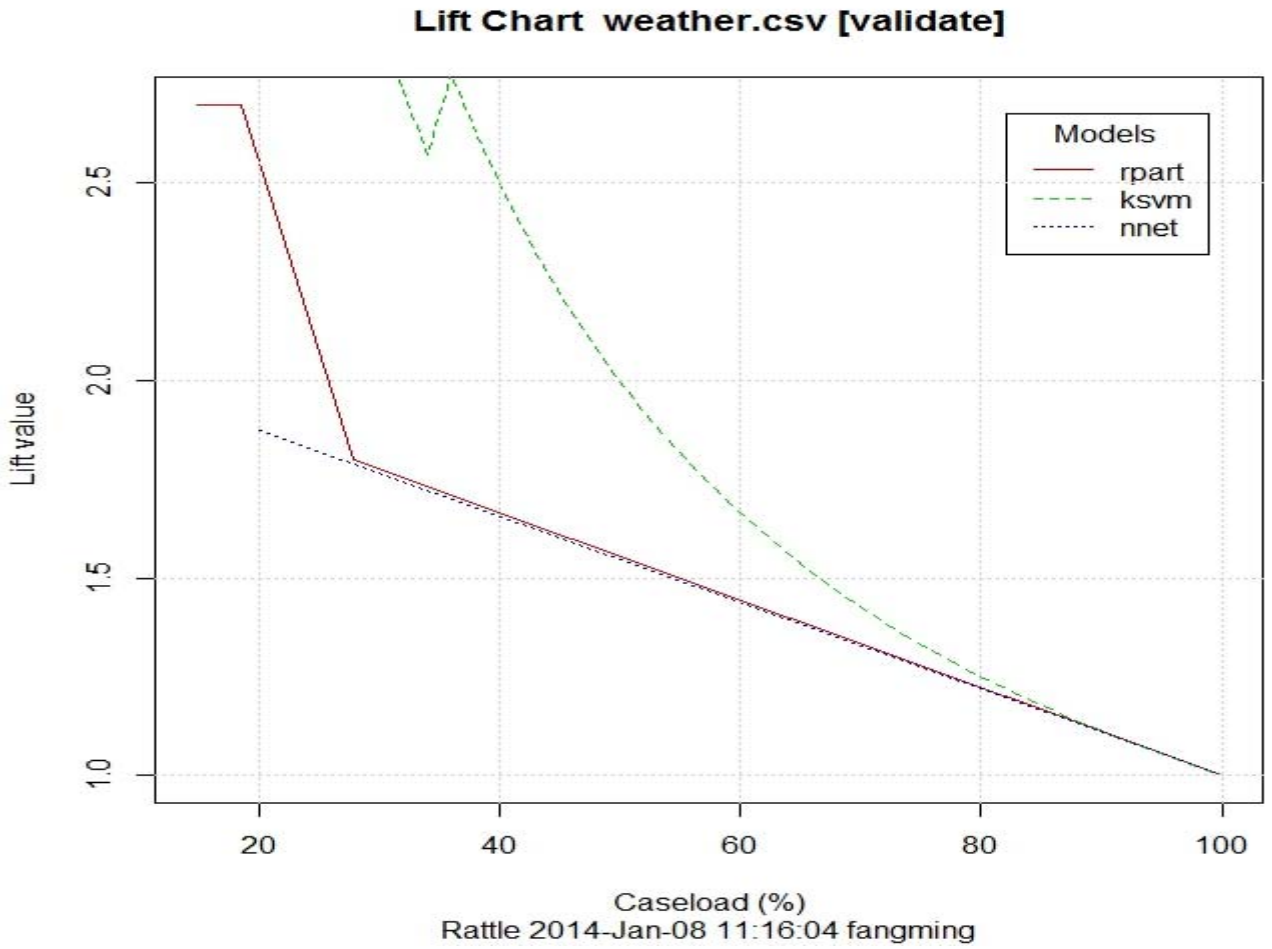
strong model

Model Assessment with Gains Chart

- Compares model performance (using Evaluation Node in SPSS Modeler)



Model Assessment with Lift Chart



Cost/Benefit Analysis

- Will the campaign yield a profit?
- Must know the campaign fixed and variable costs , e.g
  - Assume $1 to create, print & mail a brochure with the minimum number printable being 20,000.
  - Assume typical responder will spend $100 & it costs $55 to process & ship the order

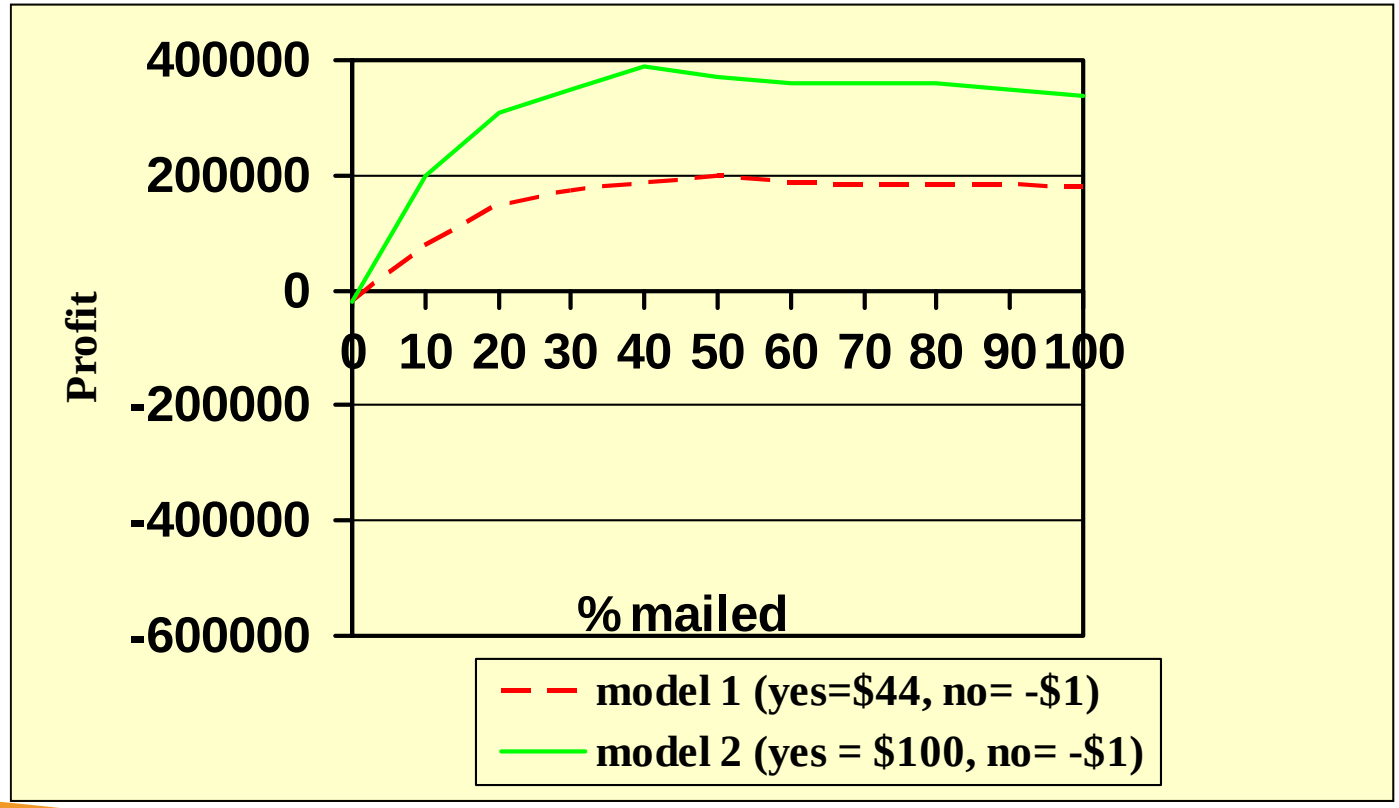
Predicted

Actual

	Yes	No
Yes	$44	$0
No	$-1	$0

Predicted Profit Chart

- Predicted Profit is sensitive to the assumed costs & the expected response rate – use with caution!

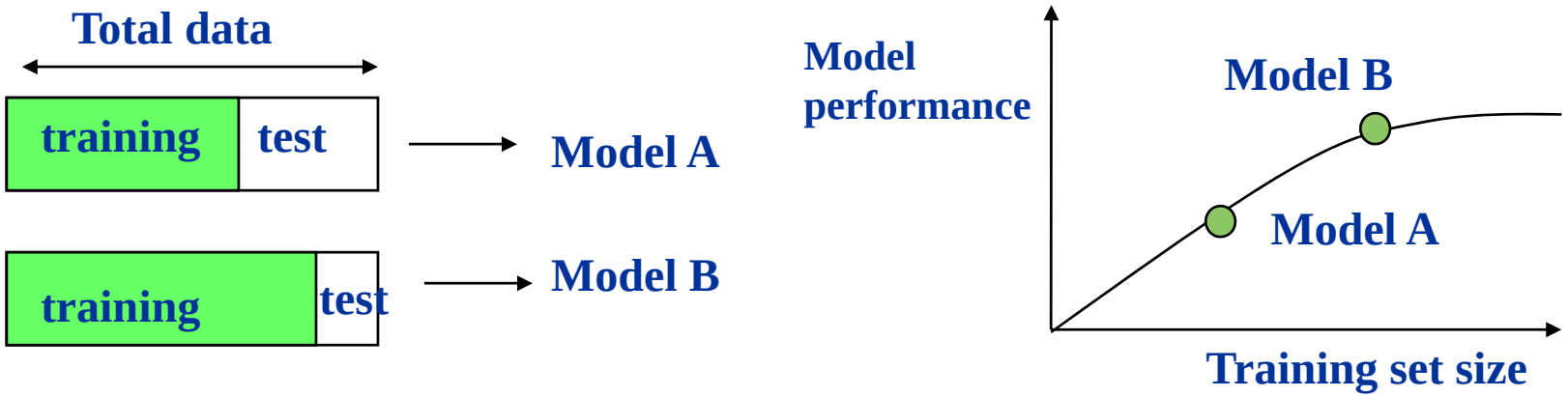


Qualitative Model Evaluation

- Inspect the models if possible, do they make sense?
- Do the importance levels they assign to the inputs agree with common sense/domain knowledge?
- Are the models stable?
- Is there evidence for overtraining?
- Are the model parameters adjusted correctly?
- Try different modeling methods
- Can a combined model improve performance?

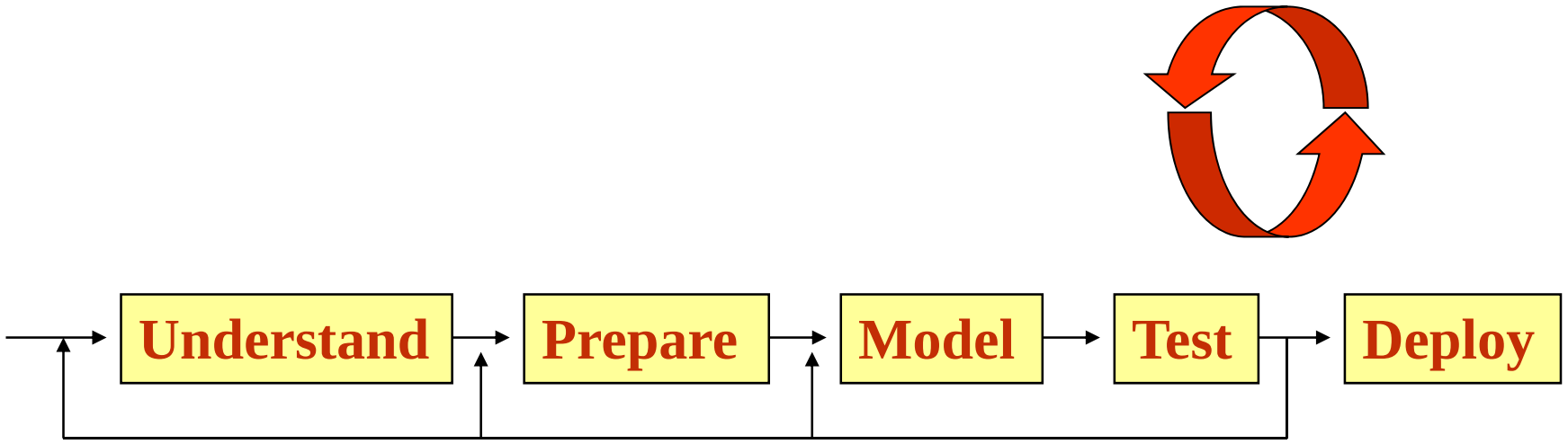
Investigate Model Stability

- In an ideal world one 'correct' model will be found – but this may require perfect training data
- In practice, changes in the training data size or content often yield different models



Can the Model be Improved?

- Adjust model parameters and iterate
- Try different modeling technique
- Improve data quality, new transformations, new and better data, etc.
- Experiment with multiple models

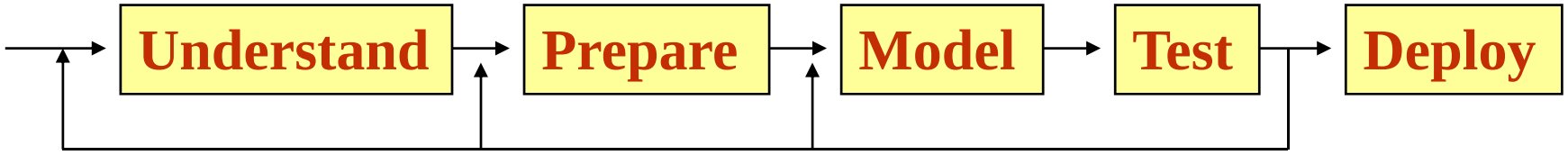


Model Deployment

- How will the models be utilised?
  - One-off use, build a mailing list, write a report, etc.
- For frequent reuse, export the model and incorporate into an operational decision support system
- Models commonly saved as:
  - C / Java code for general use
  - R script / R model object
  - Proprietary code for use by other software from same vendor
  - PMML (predictive modeling markup language) – and XML format designed to allow interchangeability of models

Summary

- Follow the methodology for successful model building



- Do not give up after building just one model
- Do not assume one test will show the true model performance
- Set a performance (or other assessment) goal for the model – stop when you have achieved it