Biostat 202: Opportunities and Challenges of "Big Data". Machine Learning 3: More Classification

John Kornak

Division of Biostatistics

Department of Epidemiology and Biostatistics

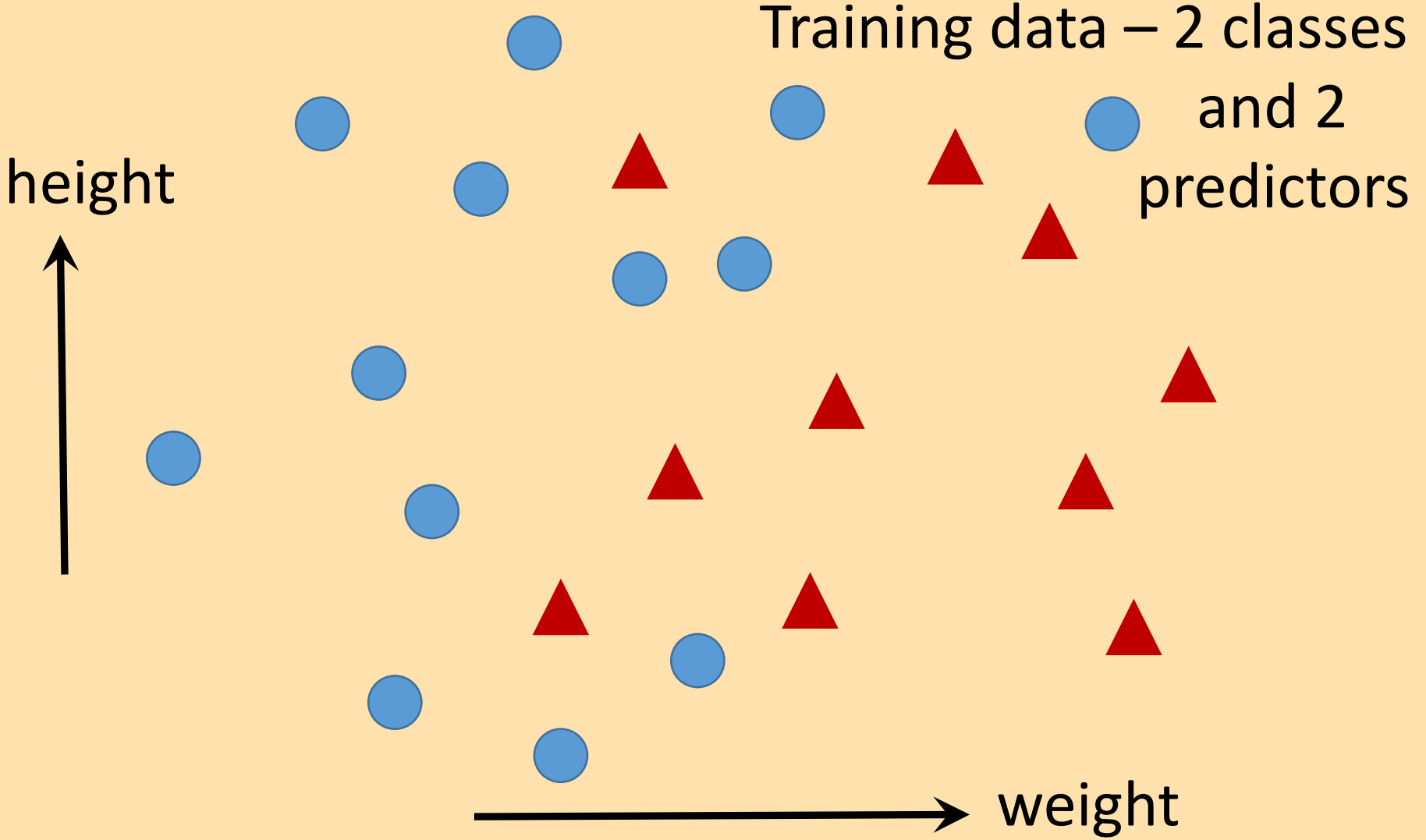
08/21/2017

Overview

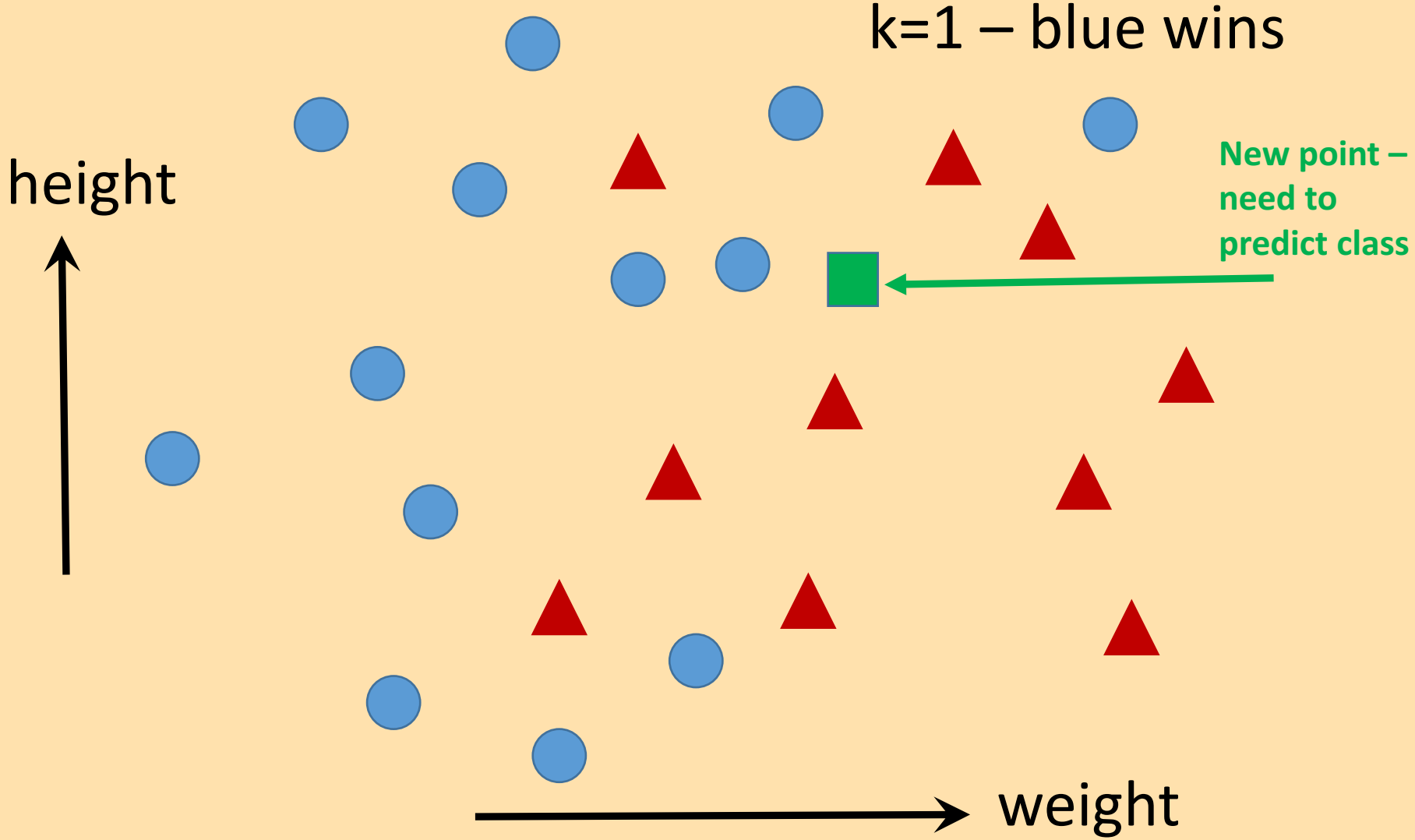
- More Classification methods
  - K-nearest neighbors
  - Classification/decision Trees (Recursive Partitioning)
  - Support vector machines
- Training-validation-test
- Cross-validation
- Comparison of classifiers

K-nearest neighbors classification

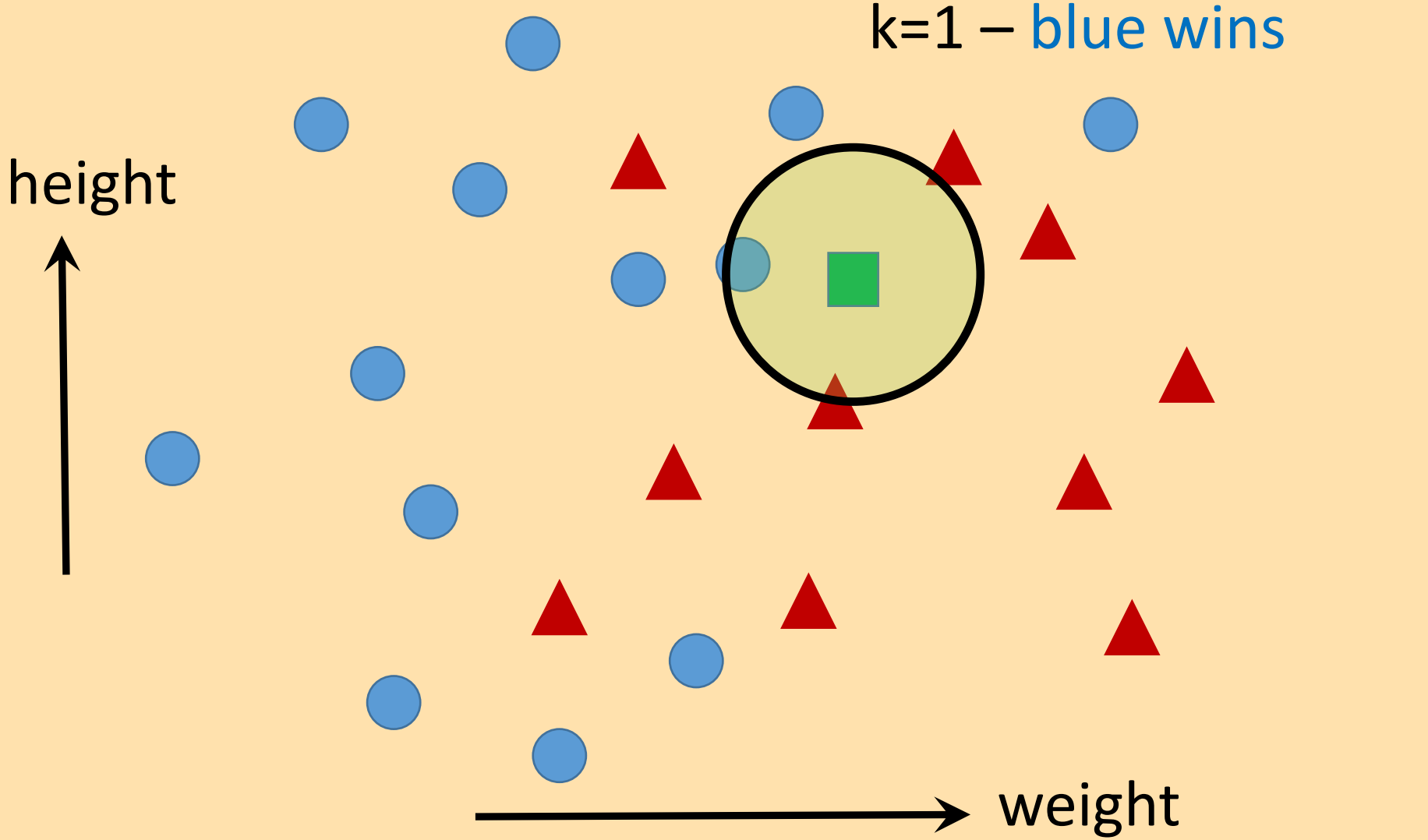
K-nearest neighbors classification



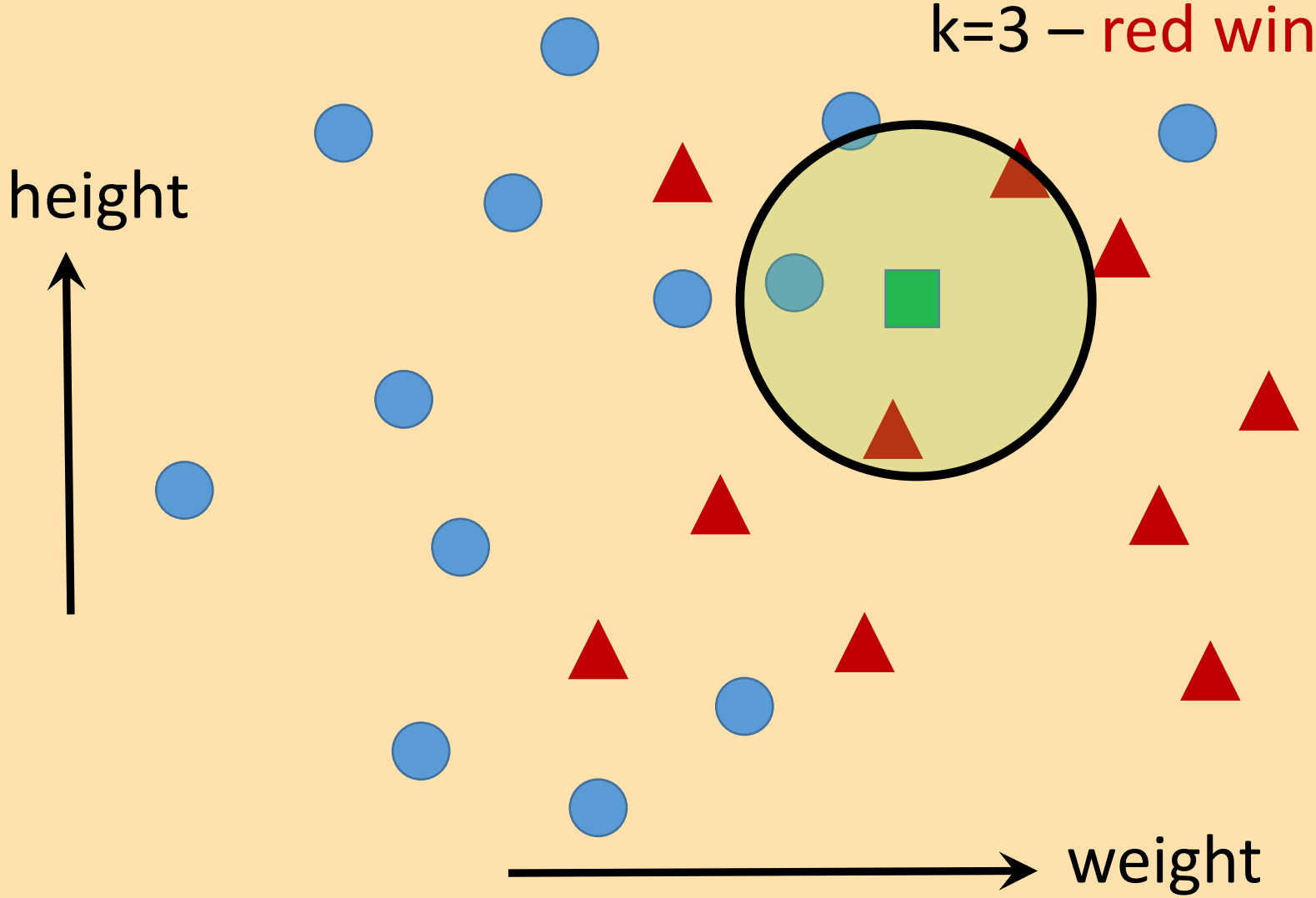
K-nearest neighbors classification



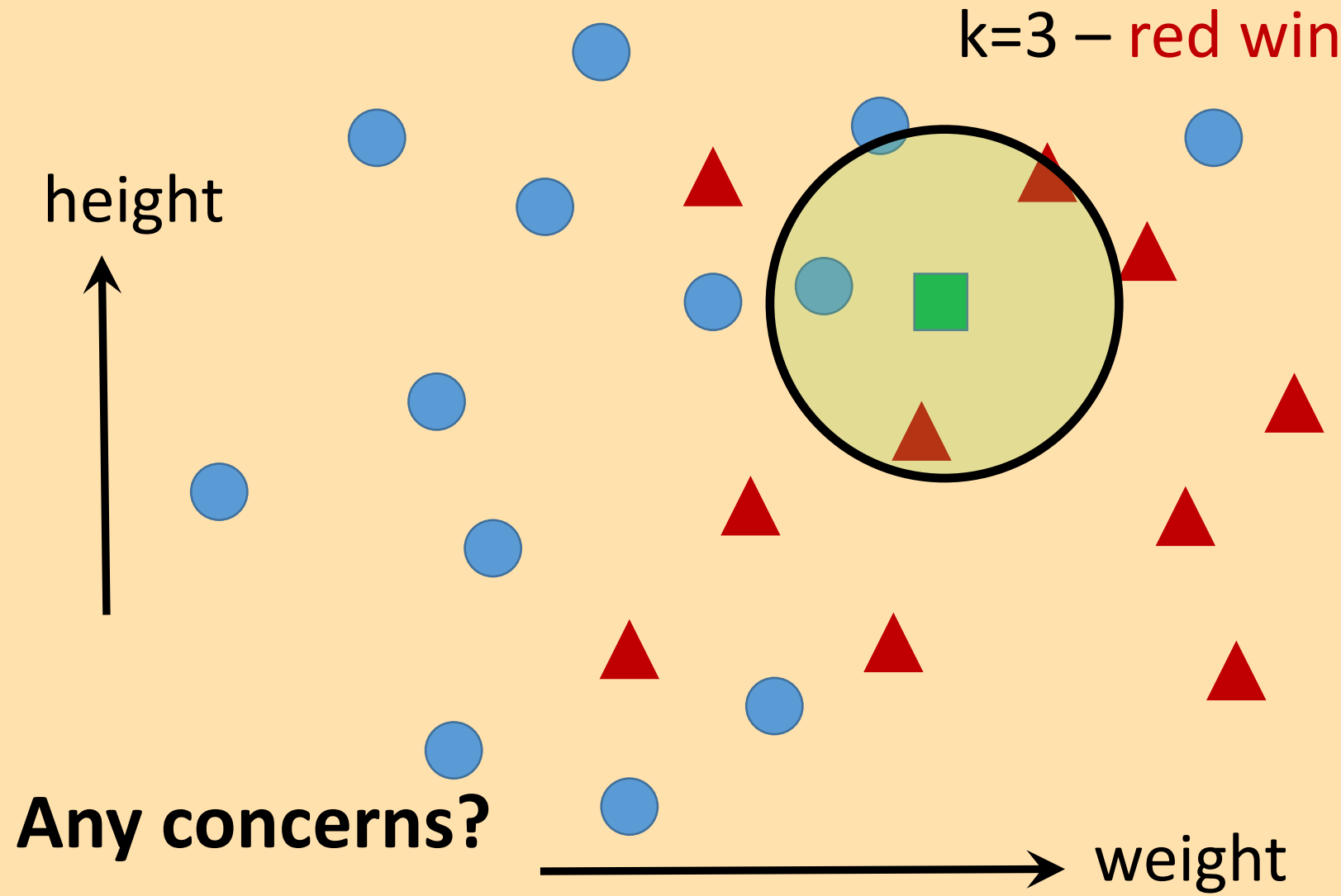
K-nearest neighbors classification



K-nearest neighbors classification



K-nearest neighbors classification



Any concerns?

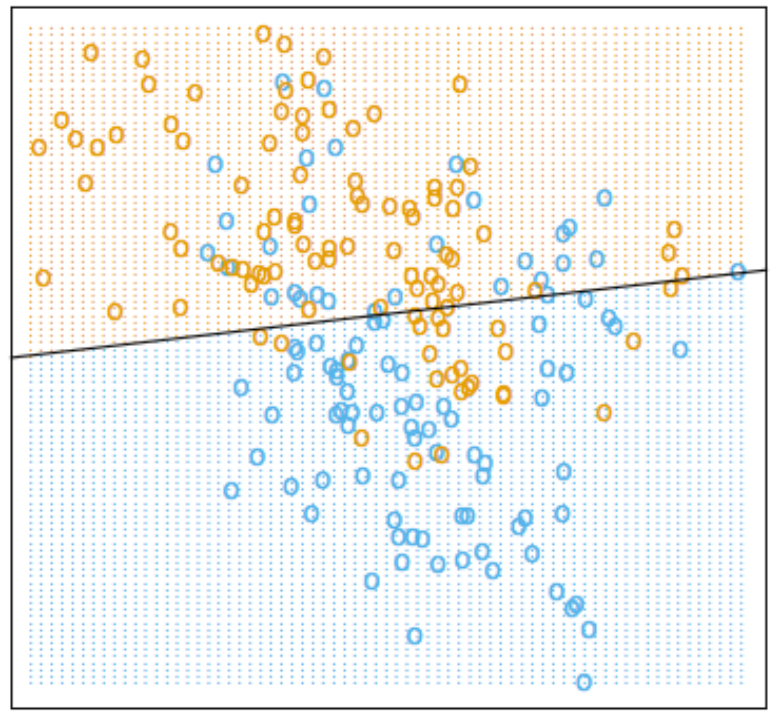
K-nearest neighbors classification

- Common to standardize predictors to have mean 0 and standard deviation of 1 (subtract mean and divide by standard deviation)
- Take majority vote among k-nearest neighbors
- Ties are broken at random (e.g. 1 red neighbor and 1 blue neighbor when k=2 – choose red or blue at random).
- Works for problems with more than 2 classes

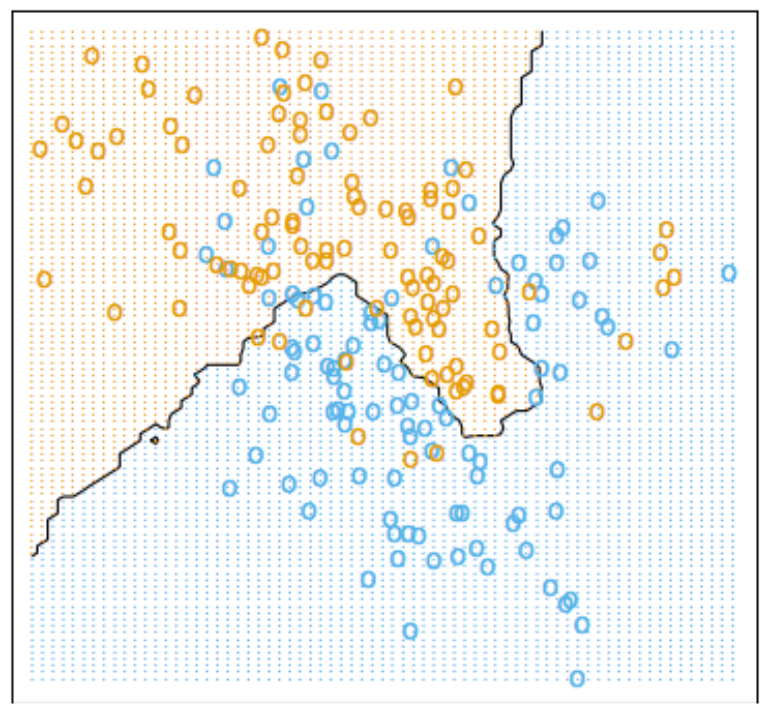
Choice of k

Example taken from Elements of Statistical Learning – Hastie et al. 2nd Ed. pp. 13-17

Linear decision boundary (LDA)

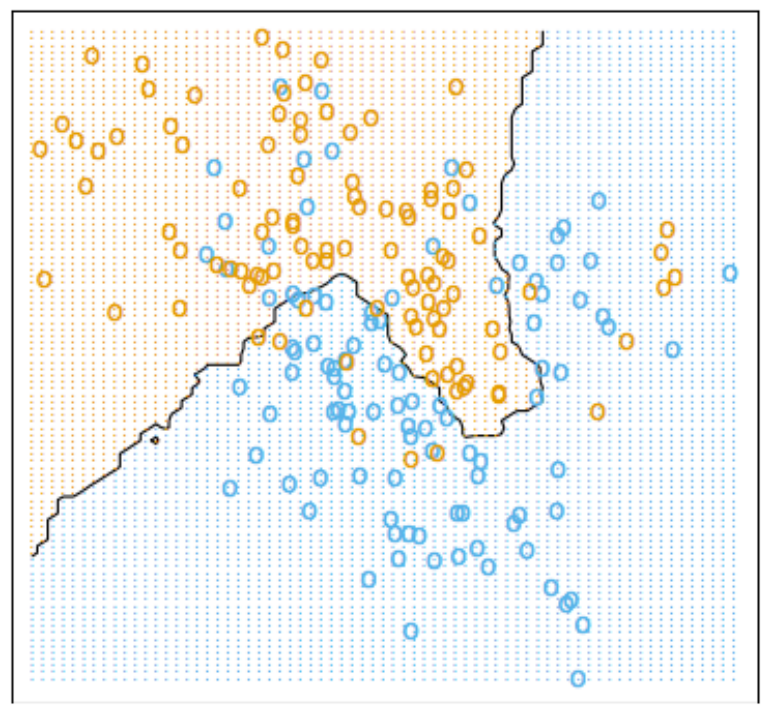


15 nearest-neighbor boundary

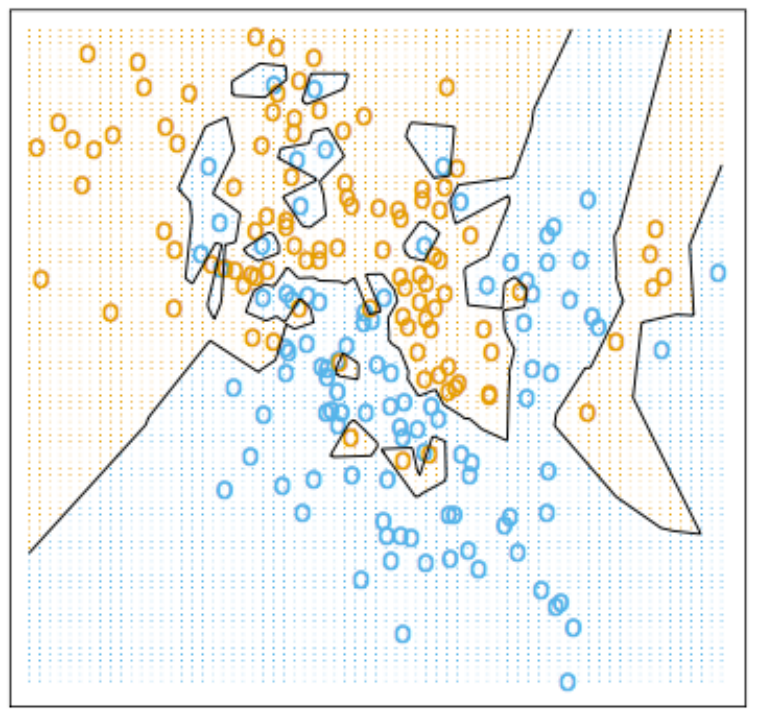


Choice of k

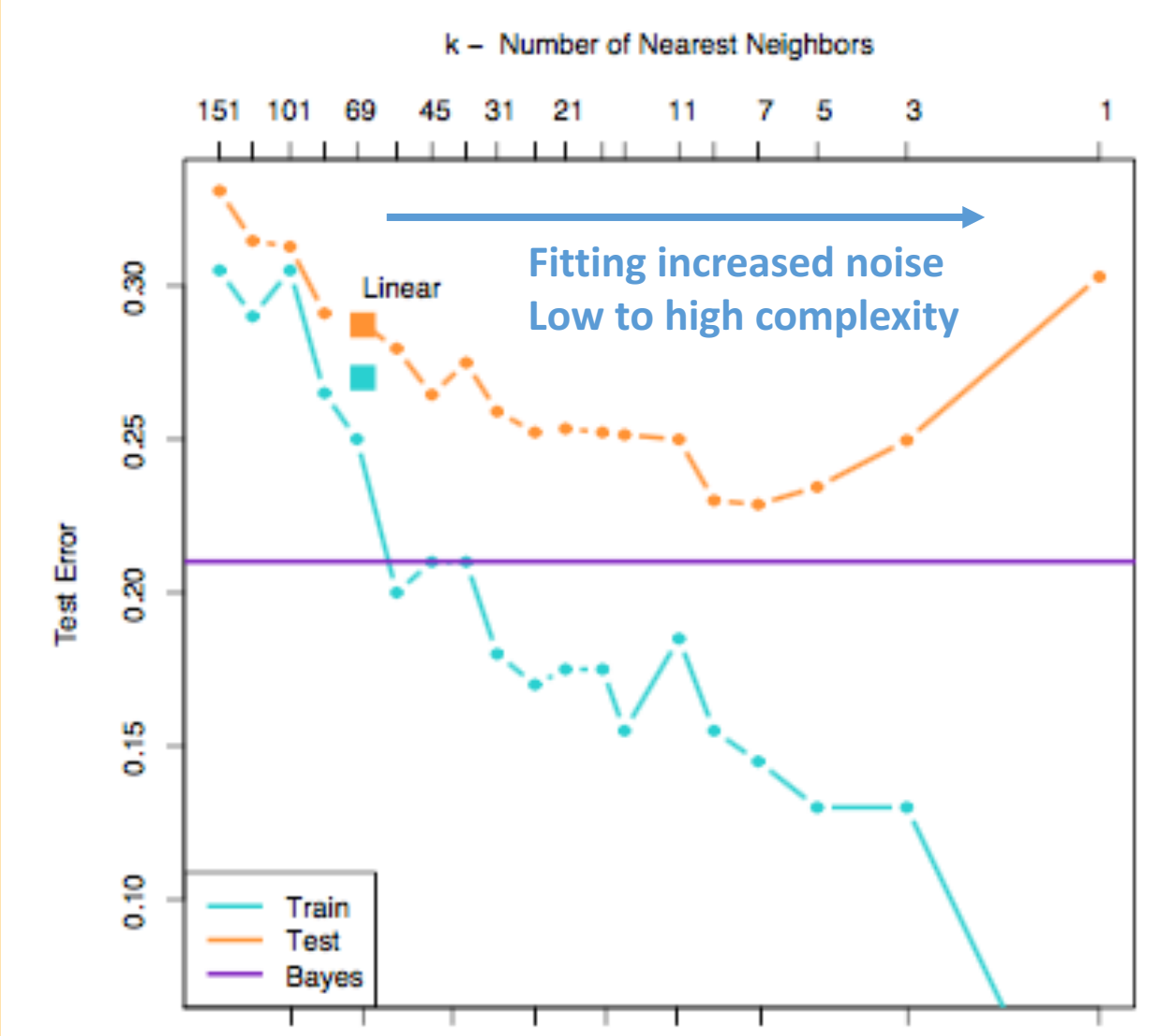
15 nearest-neighbor boundary



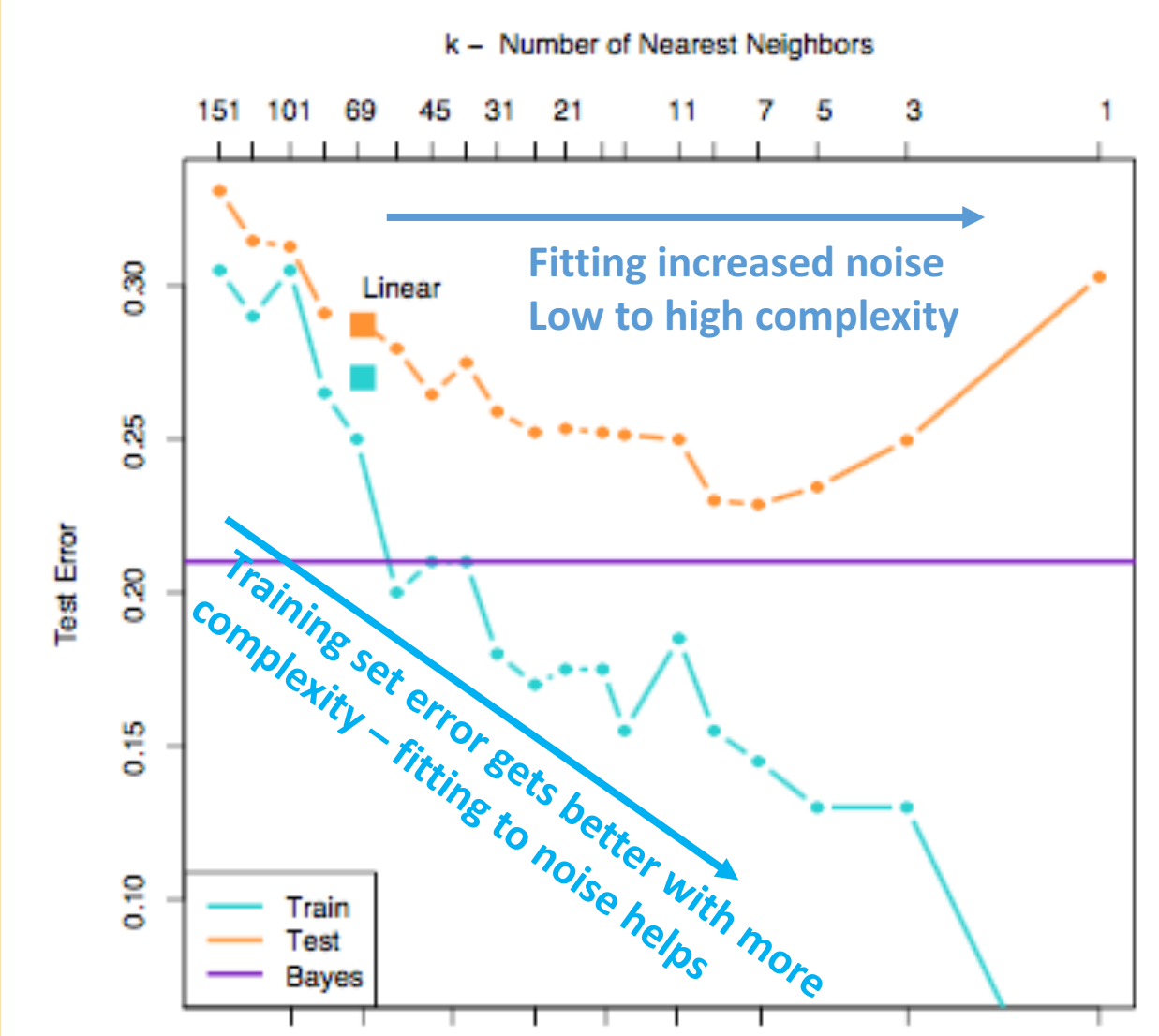
1-nearest neighbor boundary



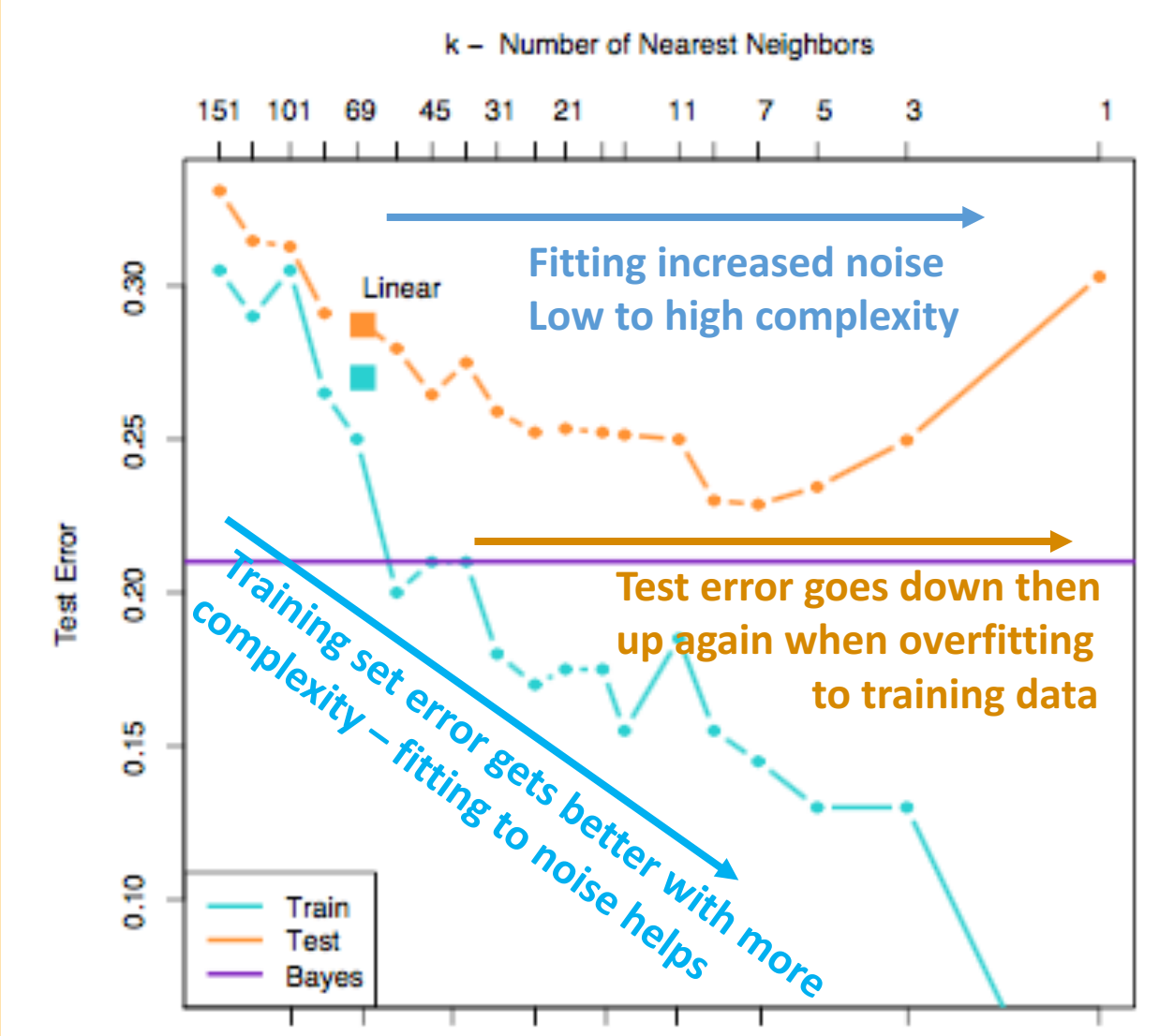
Choice of k


k – Number of Nearest Neighbors
151
101
69
45
31
21
11
7
5
3
1
Test Error
0.30
0.25
0.20
0.15
0.10
Linear
Fitting increased noise
Low to high complexity
Train
Test
Bayes

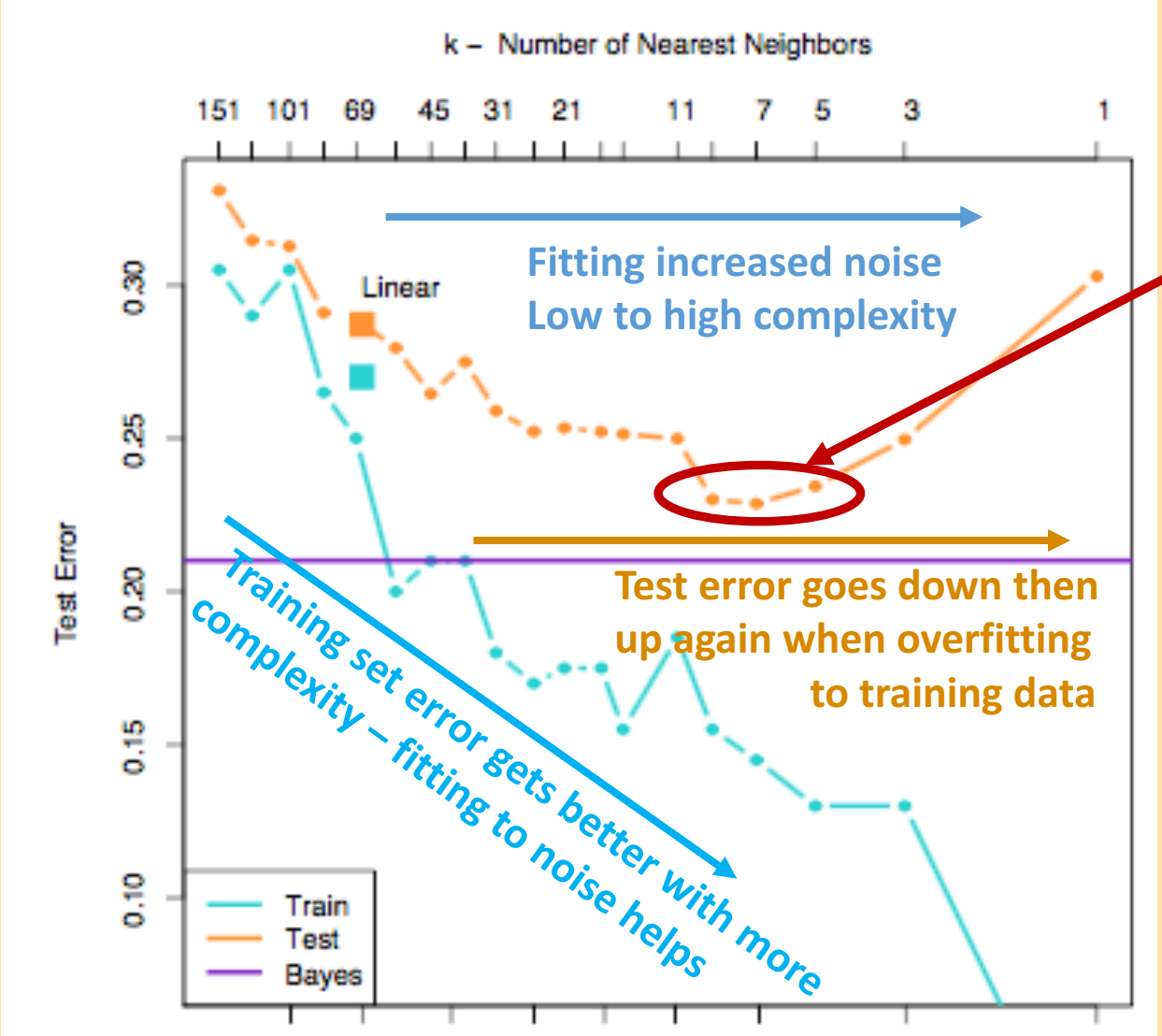
Choice of k


k – Number of Nearest Neighbors
151
101
69
45
31
21
11
7
5
3
1
Test Error
0.30
0.25
0.20
0.15
0.10
Linear
Fitting increased noise
Low to high complexity
Training set error gets better with more complexity – fitting to noise helps
Train
Test
Bayes

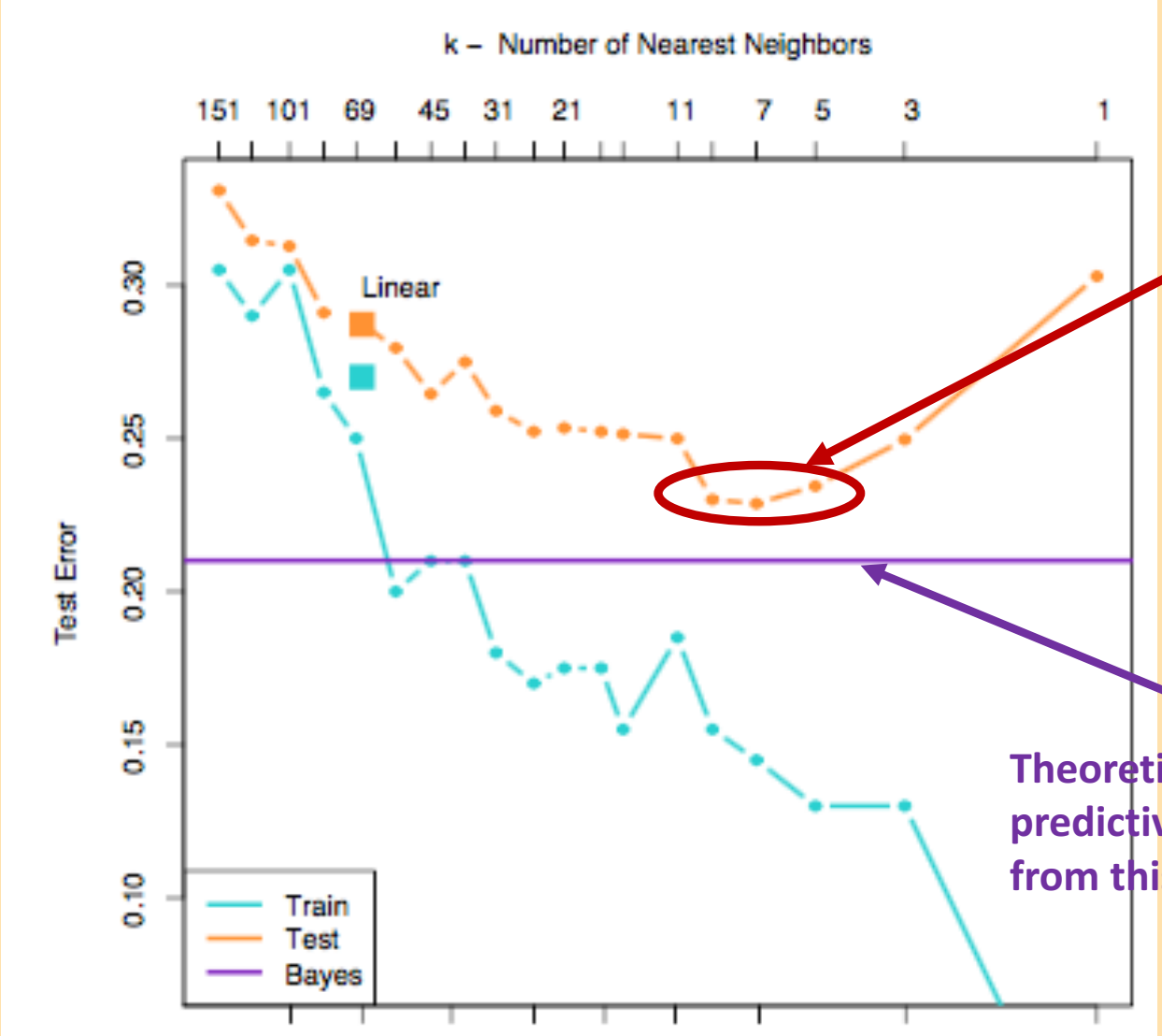
Choice of k


k – Number of Nearest Neighbors
151
101
69
45
31
21
11
7
5
3
1
Test Error
0.30
0.25
0.20
0.15
0.10
Linear
Fitting increased noise
Low to high complexity
Test error goes down then
up again when overfitting
to training data
Training set error gets better with more
complexity – fitting to noise helps
Train
Test
Bayes

Choice of k


k – Number of Nearest Neighbors
151 101 69 45 31 21 11 7 5 3 1
Test Error
0.10 0.15 0.20 0.25 0.30
Linear
Fitting increased noise
Low to high complexity
Best predictive models
Test error goes down then up again when overfitting to training data
Training set error gets better with more complexity – fitting to noise helps
Train
Test
Bayes

Choice of k

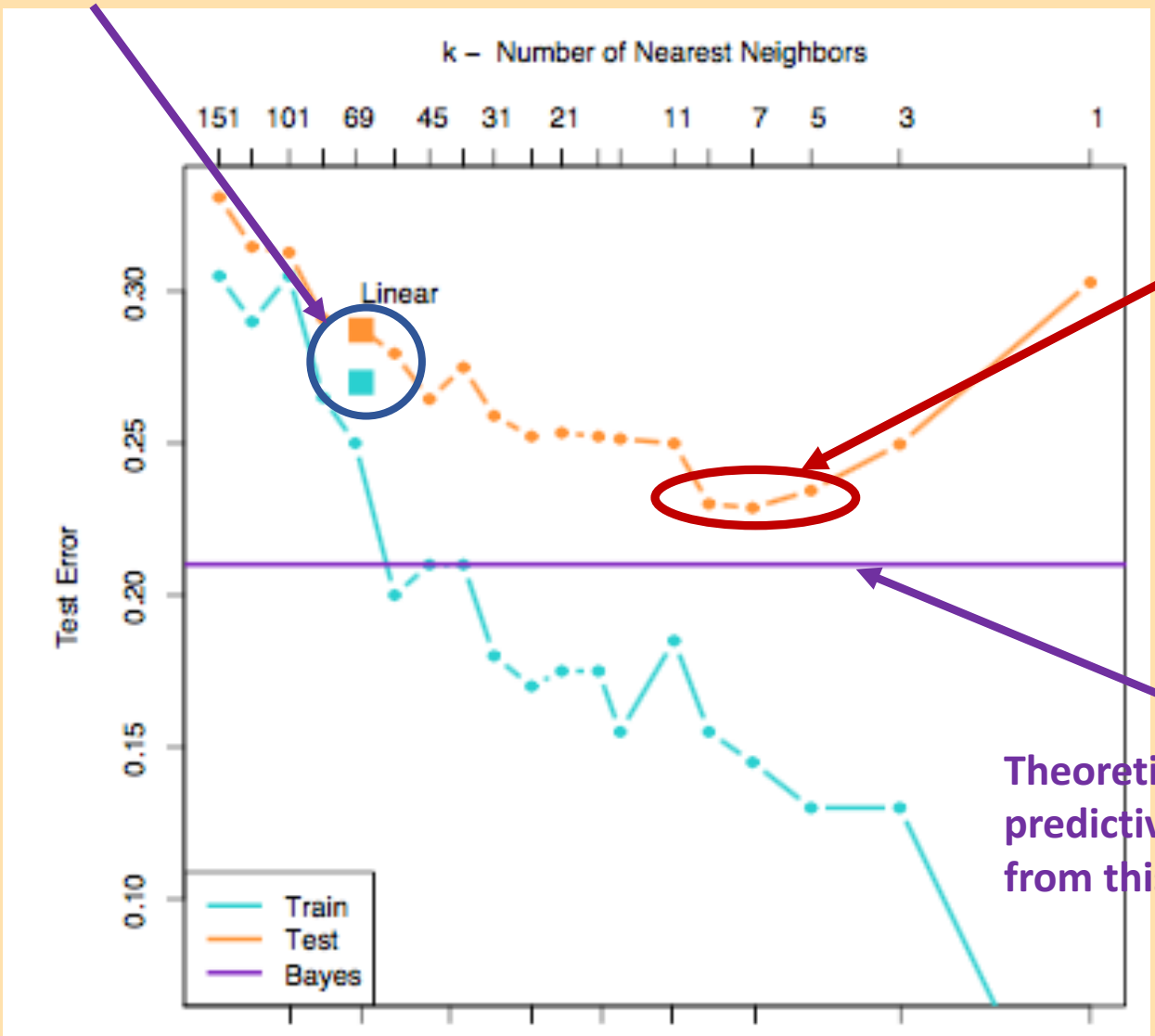


Best predictive models

Theoretically best true predictive error level from this simulated data

Choice of k

Best linear separating line



Best predictive models

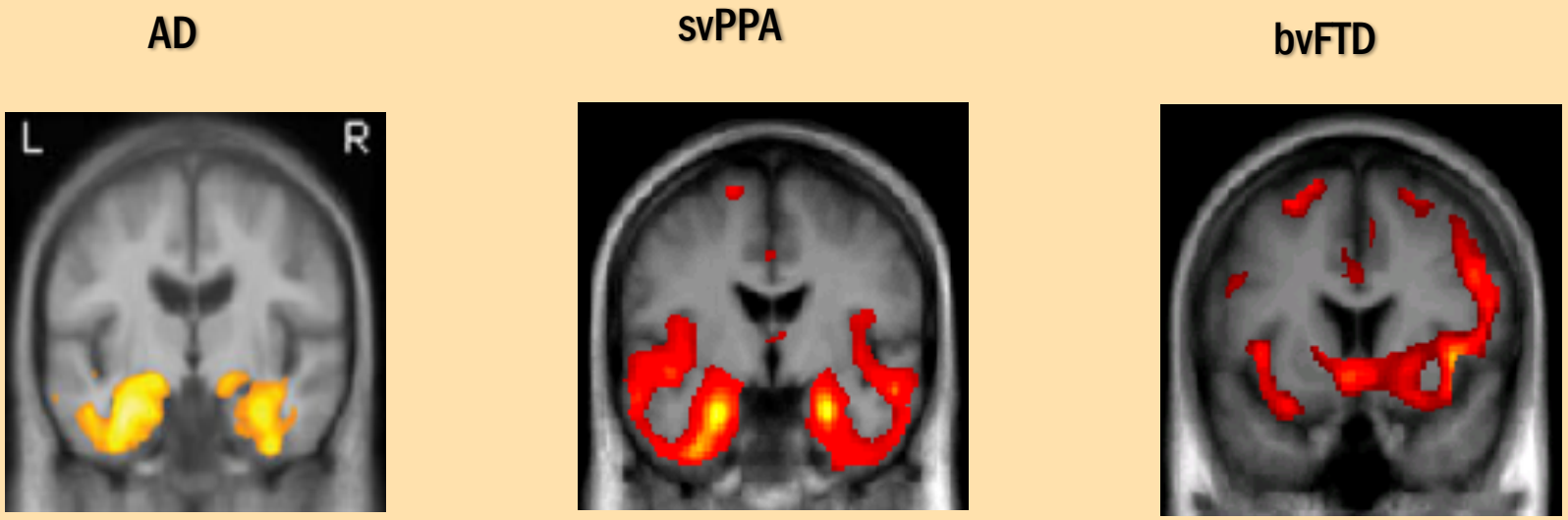
Theoretically best true predictive error level from this simulated data

Nearest neighbors classifier

- Simple technique
- No modeling
- Not good for a large number of predictors – suffers from the “curse of dimensionality”; but we can use unsupervised data reduction methods first to create new variables – e.g. Principle Components [see next lecture], or perform variable selection
- Has been applied to many machine learning problems successfully

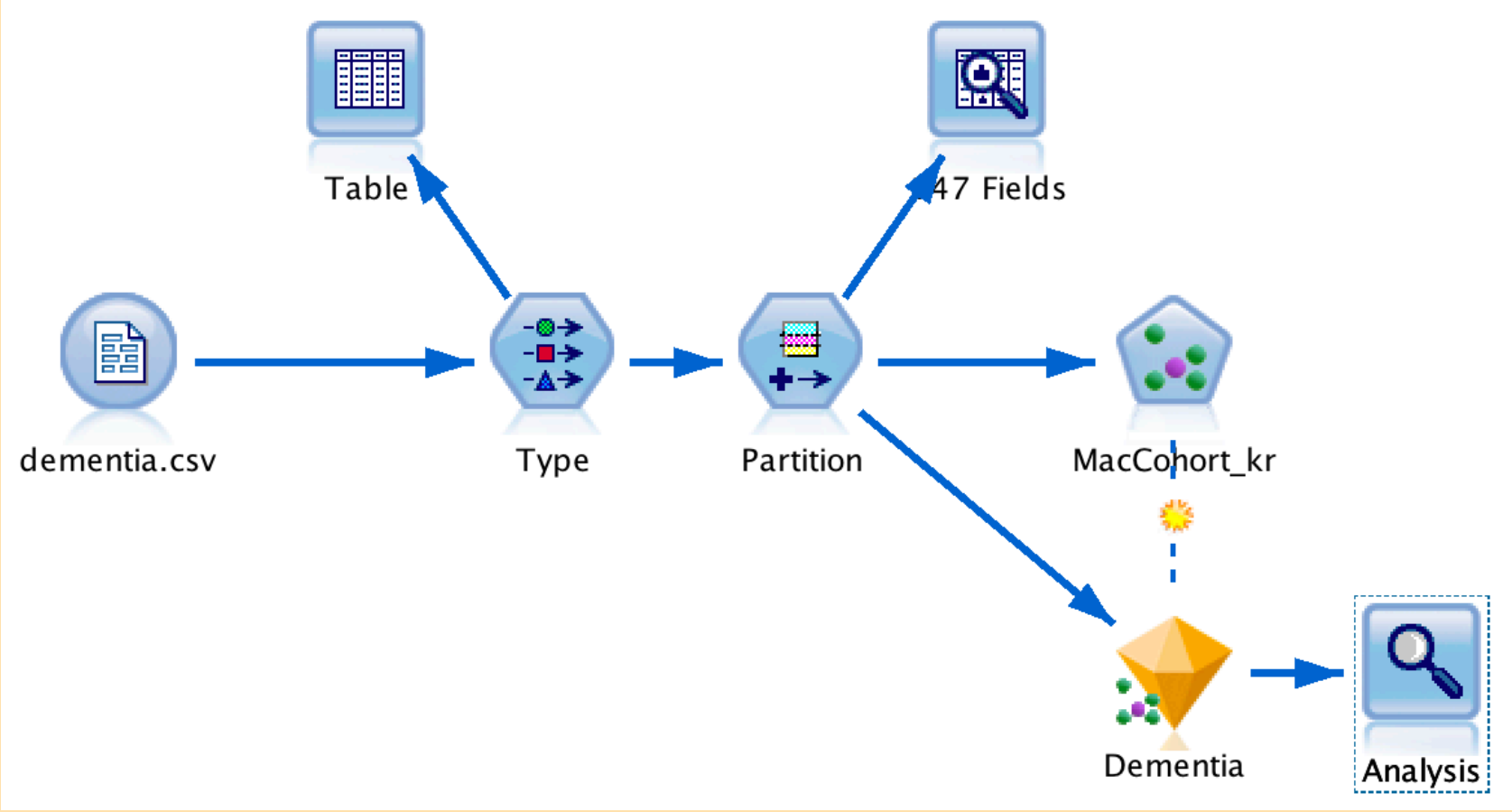
Running example: extend to differential diagnosis of multiple dementias

- Dementias have different patterns of brain loss

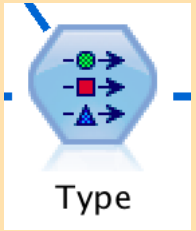


- Clinicians/radiologists often disagree on diagnosis of dementia type based on individual images
- New treatments → differential diagnosis important
- Goal: automatically predict different dementias based on patterns of brain tissue loss

Modeler: K-Nearest Neighbors



Type Node



Type

Preview

Types | Format | Annotations

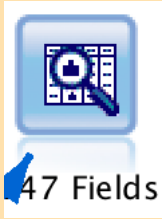
Read Values | Clear Values | Clear All Values

Field	Measurement	Values	Missing	Check	Role
field1	Continuous	[1,3095]		None	None
PIDN	Continuous	[65,19260]		None	Record ID
DCDate	Continuous	[2007-11-16,2...		None	None
SourceID	Typeless			None	None
ScanNo	Ordinal	1		None	None
vbmID	Continuous	[5,4963]		None	None
gm	Continuous	[362.558,826.9...		None	Input
wm	Continuous	[354.125,738.8...		None	Input
csf	Continuous	[189.1,515.496]		None	Input
tiv	Continuous	[1007.5,1843.11]		None	Input
MagnetStrength	Nominal	3		None	None
ageAtScan	Continuous	[33,99]		None	Input
ScannerID	Flag	"NIC 3T MRI"/"NI...		None	None
ScanType	Nominal	MP-LAS-ABN,T1...		None	None
ROWNUM	Continuous	[1,3095]		None	None
InstrID	Continuous	[126430,343119]		None	None
ScannerID_2	Flag	"NIC 3T MRI"/"NI...		None	None
MacCohort_kr	Nominal	AD,CONTROL,PS...		None	Target
PIBPos	Nominal	"0","1",NA		None	None
Sex	Flag	male/female		None	Input
Educ	Continuous	[4,32]		None	Input
Race	Nominal	"Asian Indian","B...		None	Input
DateofDeath	Continuous	[2010-06-11,2...		None	None
autopsy.	Nominal	"","AUTOPSY PE...		None	None
PIBDone.	Flag	Yes/No		None	None
X3rd_Ventricle	Continuous	[58.75655516,1		None	Input

View current fields | View unused field settings

OK | Cancel | Apply | Reset

Audit Node (post-Type)



Data Audit of [147 fields]

File Edit Generate

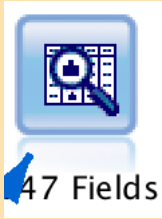
Audit Quality Annotations

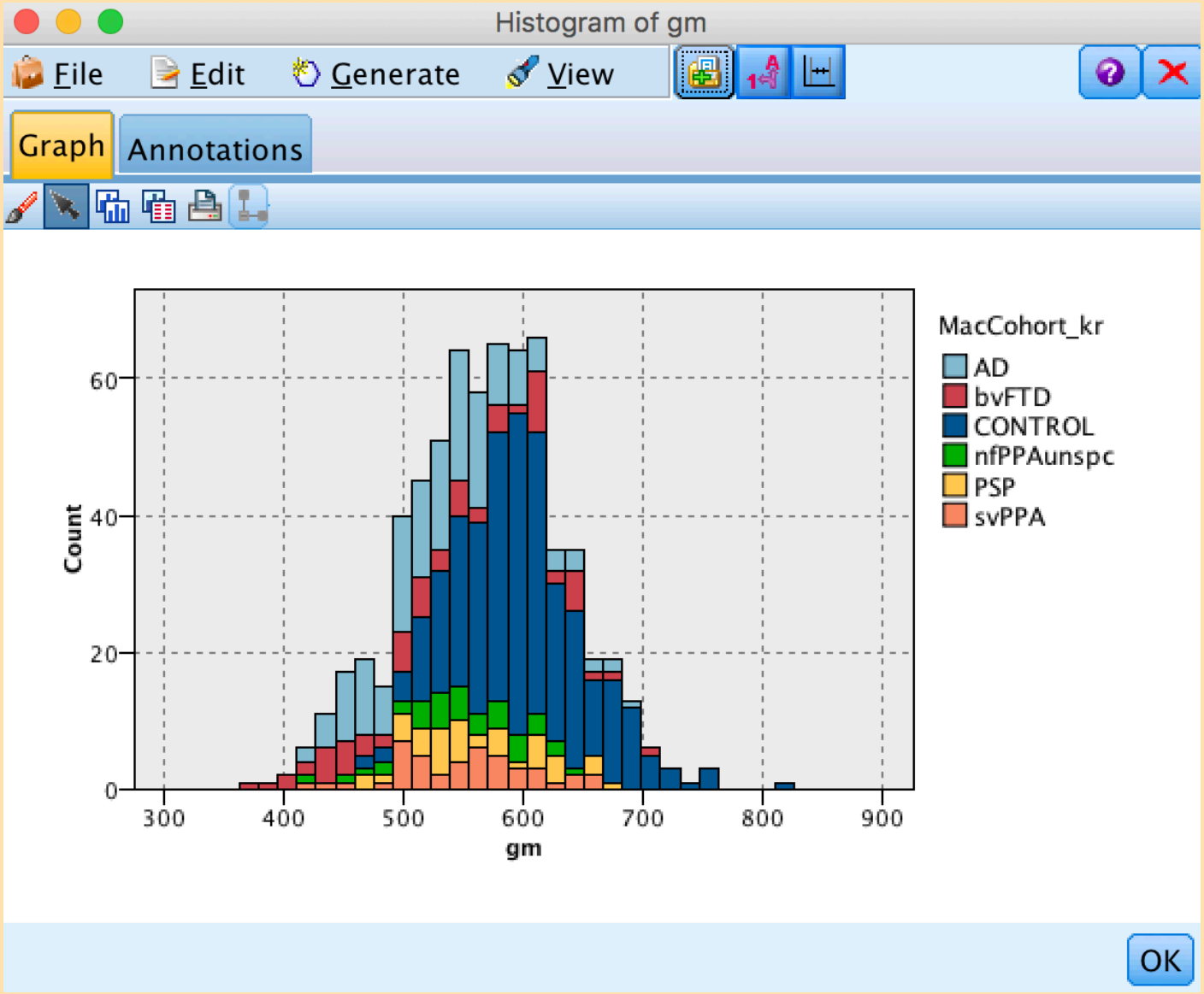
Field	Graph	Measurement	Min	Max	Sum	Range	Mean	Mean Std. Err.	Std. Dev	Variance	Skewness	Skewness Std. Err.	Kurtosis
PIDN		Continuous	65	19260	8165983	19195	12372.7...	136.789	3514.176	12349432....	-0.021	0.095	-0.96
gm		Continuous	62.558	826.936	375347.105	464.378	568.708	2.568	65.967	4351.623	0.031	0.095	0.19
wm		Continuous	354.125	738.869	325321.382	384.744	492.911	2.377	61.072	3729.795	0.478	0.095	0.31
csf		Continuous	189.100	515.496	198180.997	326.396	300.274	2.156	55.397	3068.854	0.496	0.095	0.09
tiv		Continuous	1007.500	1843.110	898849.600	835.610	1361.893	5.175	132.946	17674.516	0.437	0.095	0.22
ageAtScan		Continuous	33	99	44021	66	66.698	0.348	8.945	80.014	0.082	0.095	0.02
MacCohort_kr		Nominal	--	--	--	--	--	--	--	--	--	--	
Sex		Flag	--	--	--	--	--	--	--	--	--	--	

[1] Indicates a multimode result [2] Indicates a sampled result

OK

Audit Node (post-Type)





KNN node

MacCohort_kr

Objectives | Fields | Settings | Annotations

The kNN procedure will identify the most similiar training cases (the nearest neighbors) to your cases of interest. A target field can be predicted based on the neighboring values.

What type of analysis do you want to perform?

- (•) Predict a target field
- ( ) Only identify the nearest neighbors

What is your objective?

- (•) Balance speed and accuracy
  Automatically selects the best number of neighbors within a small range.
- ( ) Speed
  Finds a fixed number of neighbors.
- ( ) Accuracy
  Automatically selects the best number of neighbors within a larger range and uses predictor importance when calculating distances.
- ( ) Custom analysis
  Choose this option to fine tune the algorithm on the Settings tab.

OK | Run | Cancel | Apply | Reset

KNN node

MacCohort_kr

Objectives | Fields | Settings | Annotations

Use predefined roles

Use custom field assignments

Target: MacCohort_kr

Inputs:

- wm
- X3rd_Ventricle
- X4th_Ventricle
- Right_Accumbens_Area
- Left_Accumbens_Area
- Right_Amygdala
- Left_Amygdala
- Brain_Stem
- Right_Caudate
- Left_Caudate
- Right_Cerebellum_Exterior
- Left_Cerebellum_Exterior
- Right_Cerebellum_White_Matter
- Left_Cerebellum_White_Matter
- Right_Cerebral_White_Matter
- Left_Cerebral_White_Matter
- Cerebrospinal_Fluid
- Right_Hippocampus
- Left_Hippocampus
- Right_Inf_Lat_Vent
- Left_Inf_Lat_Vent
- Right_Lateral_Ventricle
- Left_Lateral_Ventricle
- Right_Pallidum
- Left_Pallidum

OK | Run | Cancel | Apply | Reset

KNN node

MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

Model name: ◉ Auto ○ Custom

☑ Use partitioned data

☐ Build model for each split

To select fields manually, choose "Use custom settings" on the Fields tab

Partition:

Splits:

☑ Normalize range inputs

☐ Use case labels — Sex

☐ Identify focal record — Sex

OK | Run | Cancel | Apply | Reset

KNN node

MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

Number of Nearest Neighbors (k)

- ○ Specify fixed K
  - K: 3
- ◉ Automatically select k
  - Minimum: 3
  - Maximum: 7

Distance Computation

- ◉ Euclidean metric
- ○ City Block metric

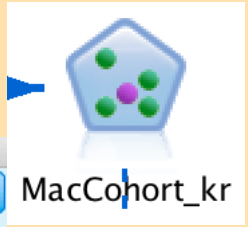
☑ Weight features by importance when computing distances

Predictions for Range Target

- ◉ Mean of nearest neighbor values
- ○ Median of nearest neighbor values

OK | Run | Cancel | Apply | Reset

KNN node



MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

☑ Perform feature selection

Forward selection is used to evaluate features for inclusion. To force a feature into the model prior to forward selection, select it in the Forced entry list.

Forced entry:

Stopping Criterion

◉ Stop when the specified number of features have been selected

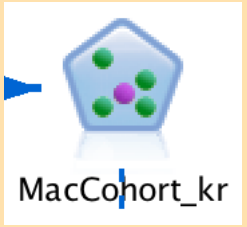
Number to select: 10

○ Stop when the change in the absolute error ratio is less than or equal to the minimum

Minimum change: 0.010

OK | Run | Cancel | Apply | Reset

KNN node



MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

Cross-Validation Folds

V-fold cross-validation is performed if you choose automatic k selection but do not choose feature selection.

Randomly assign cases to folds

Number of folds: 10

Set random seed Seed: 12345

Use field to assign cases

OK | Run | Cancel | Apply | Reset

KNN node

MacCohort_kr

Objectives | Fields | Settings | Annotations

Model
Neighbors
Feature Selection
Cross-Validation
Analyze

☑ Append all probabilities
☑ Save distances between cases and k nearest neighbors

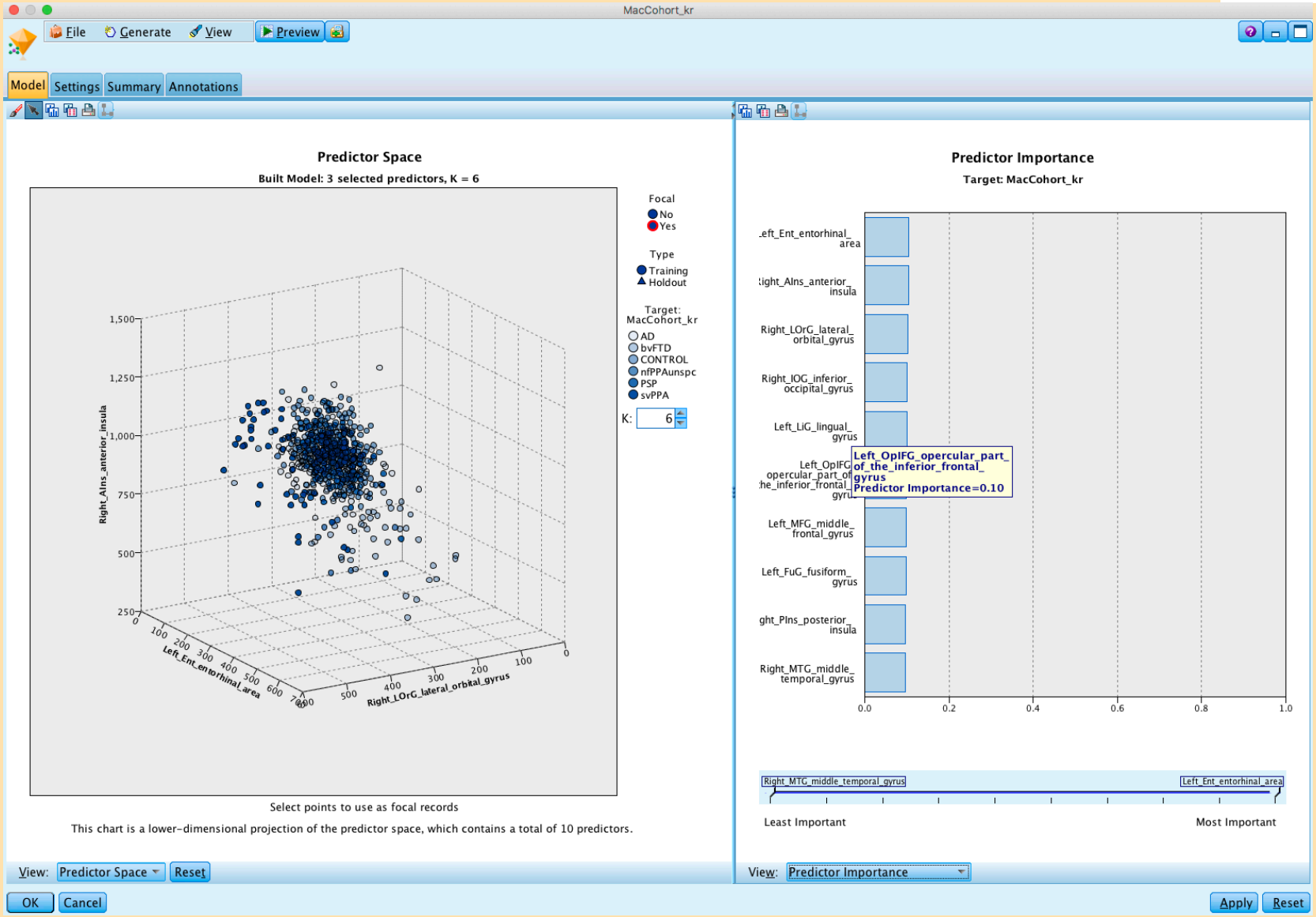
Propensity Scores (valid only for flag targets)

☐ Calculate raw propensity scores
☐ Calculate adjusted propensity scores

Based on ◉ Testing partition ◎ Validation partition

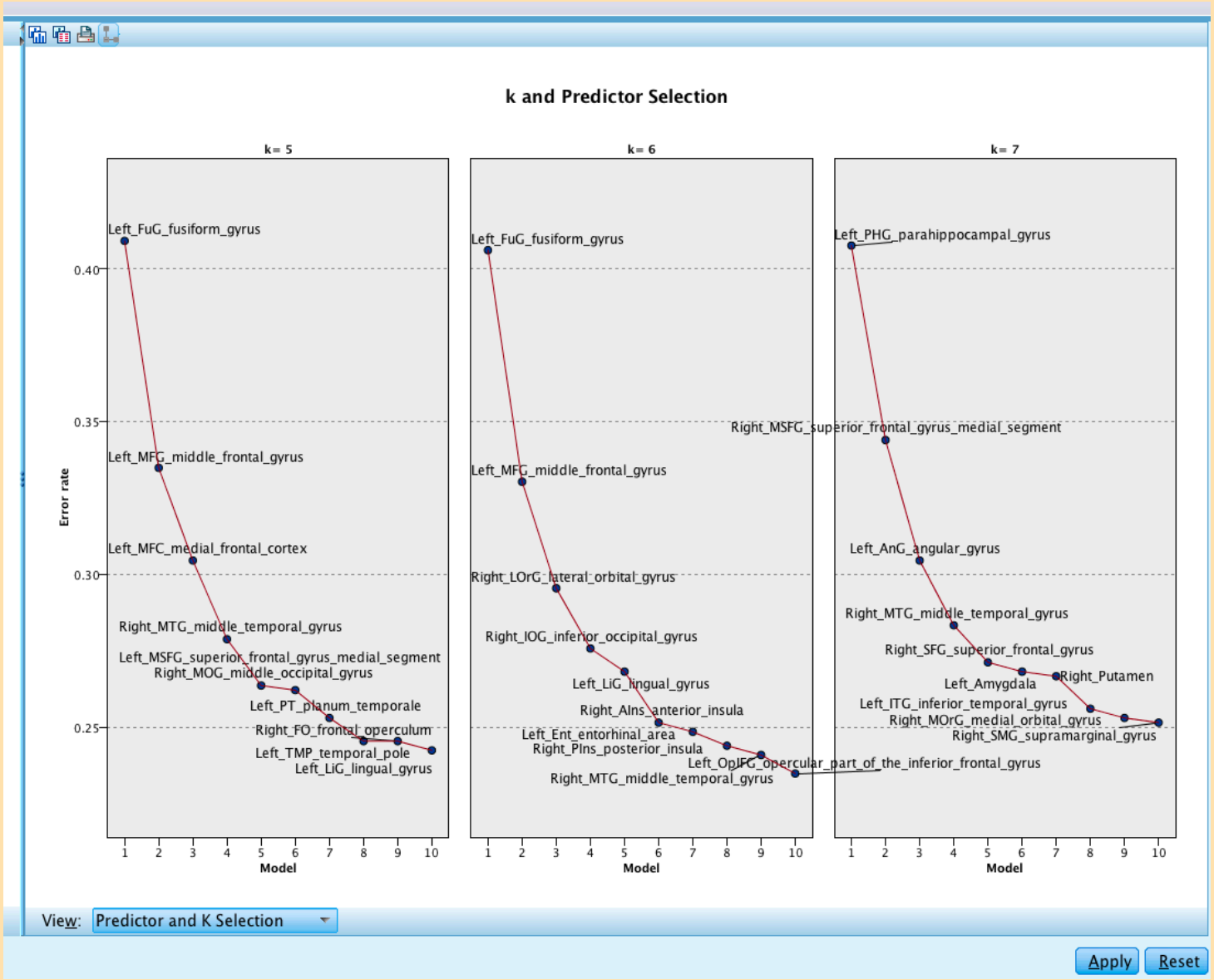
OK | Run | Cancel | Apply | Reset

Golden nugget

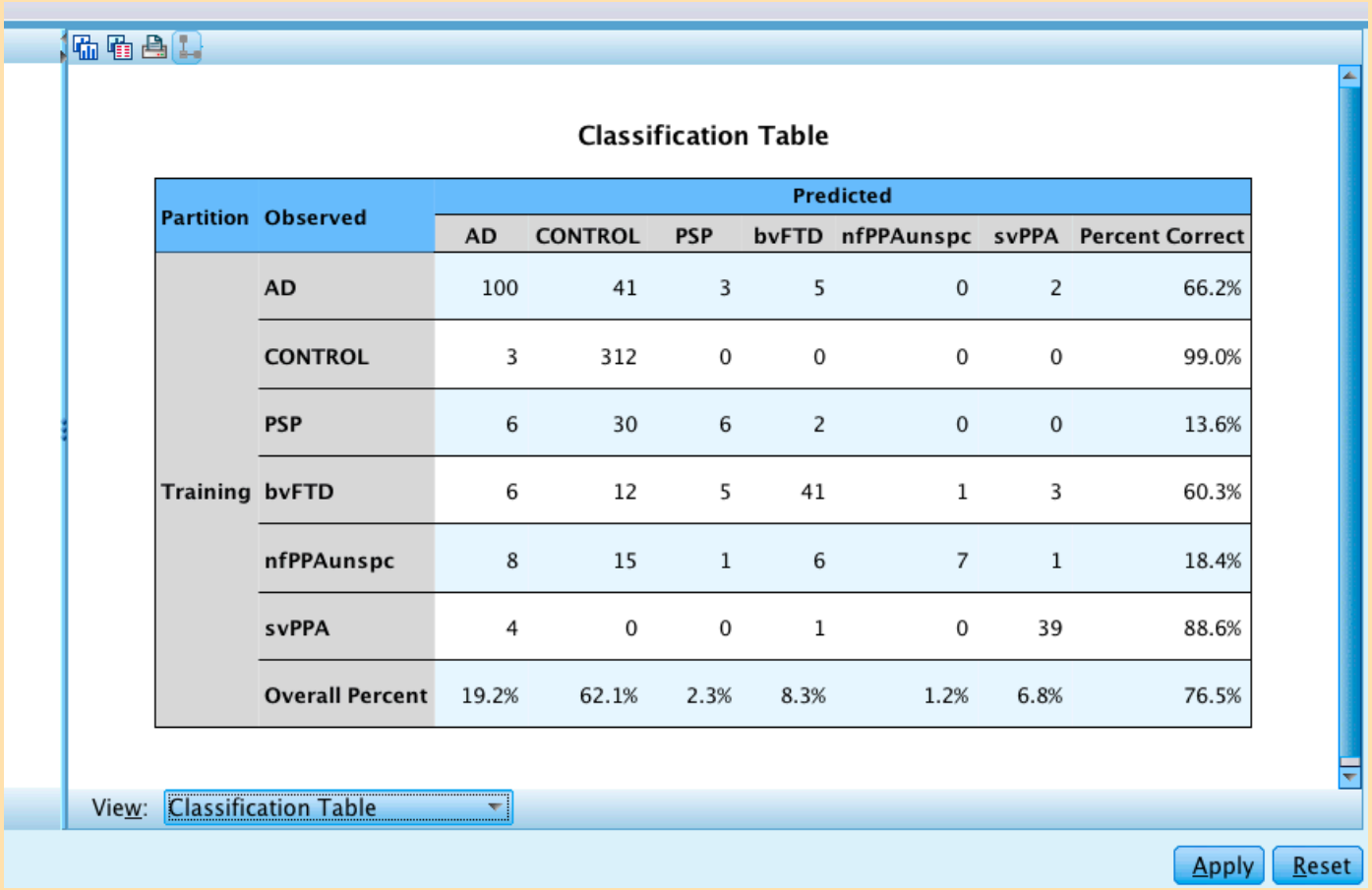


Golden nugget

MacCohort_kr



Golden nugget



Classification Table

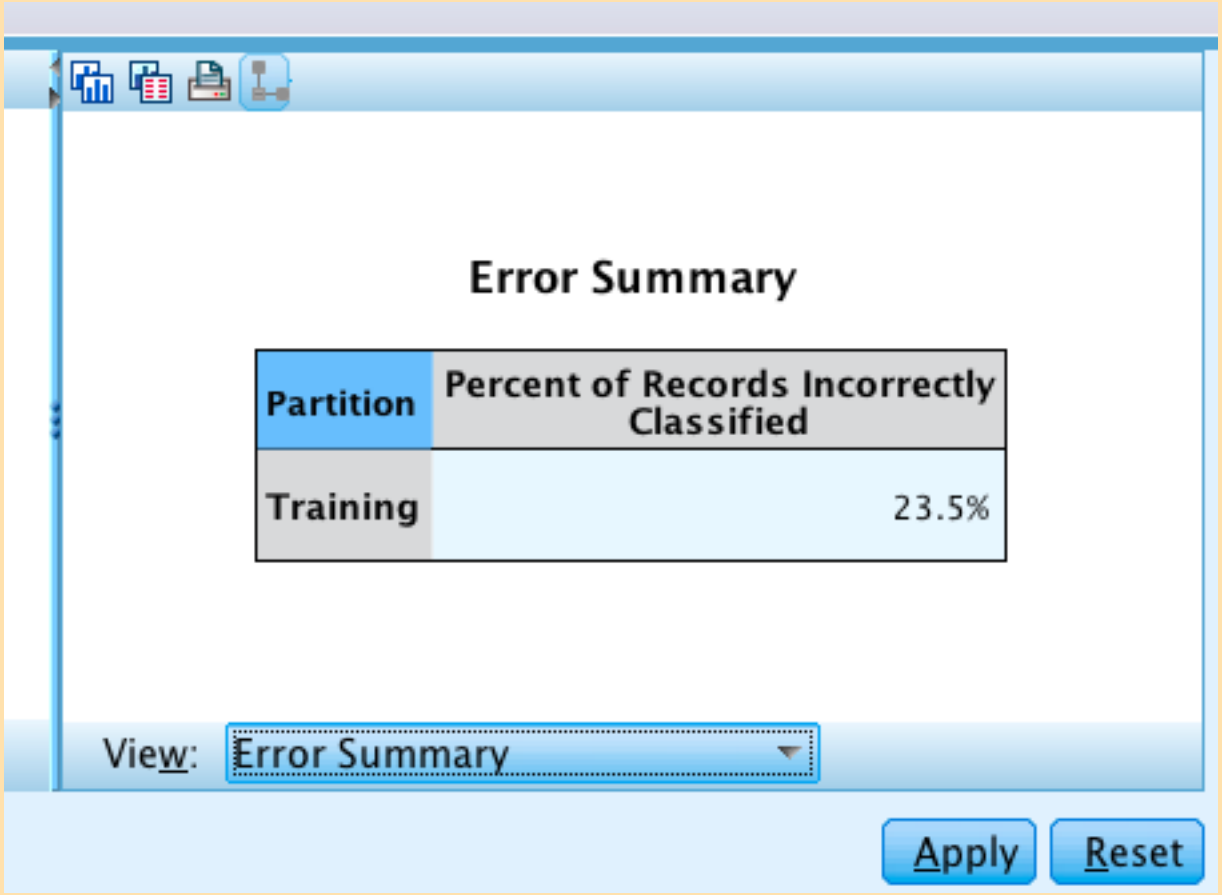
Partition	Observed	Predicted						
		AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA	Percent Correct
Training	AD	100	41	3	5	0	2	66.2%
	CONTROL	3	312	0	0	0	0	99.0%
	PSP	6	30	6	2	0	0	13.6%
	bvFTD	6	12	5	41	1	3	60.3%
	nfPPAunspc	8	15	1	6	7	1	18.4%
	svPPA	4	0	0	1	0	39	88.6%
	Overall Percent	19.2%	62.1%	2.3%	8.3%	1.2%	6.8%	76.5%

View: Classification Table

Apply Reset

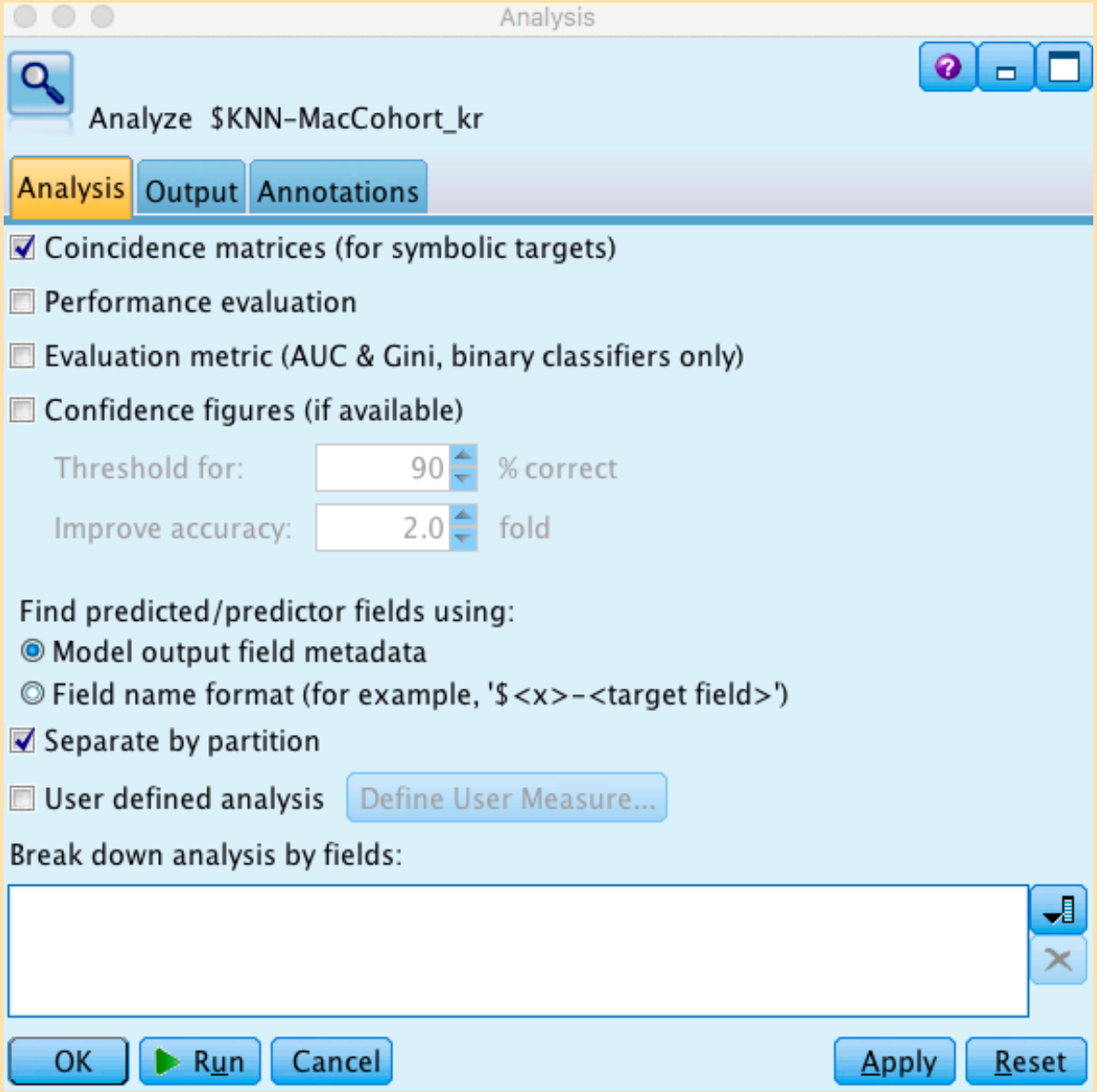
Warning: this is based on training and test combined

Golden nugget



Warning: this is based on training and test combined

Analysis node



Analysis node

Analysis of [MacCohort_kr] #1

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	331	76.8%	190	82.97%
Wrong	100	23.2%	39	17.03%
Total	431		229	

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	63	29	1	1	0	2
CONTROL	1	203	0	0	0	0
PSP	3	23	4	1	0	0
bvFTD	4	8	3	27	1	0
nfPPAunspc	8	7	0	2	6	1
svPPA	4	0	0	1	0	28

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	45	9	1	0	0	0
CONTROL	1	110	0	0	0	0
PSP	2	7	4	0	0	0
bvFTD	3	3	1	18	0	0
nfPPAunspc	0	7	1	4	2	0
svPPA	0	0	0	0	0	11

OK

Analysis node

Analysis of [MacCohort_kr] #1

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	331	76.8%	190	82.97%
Wrong	100	23.2%	39	17.03%
Total	431		229	

Anything unexpected?

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	63	29	1	1	0	2
CONTROL	1	203	0	0	0	0
PSP	3	23	4	1	0	0
bvFTD	4	8	3	27	1	0
nfPPAunspc	8	7	0	2	6	1
svPPA	4	0	0	1	0	28

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	45	9	1	0	0	0
CONTROL	1	110	0	0	0	0
PSP	2	7	4	0	0	0
bvFTD	3	3	1	18	0	0
nfPPAunspc	0	7	1	4	2	0
svPPA	0	0	0	0	0	11

OK

Different Partition?

Partition

Generate Preview

Settings Annotations

Partition field: Partition

Partitions: Train and test Train, test and validation

Training partition size: 65 Label: Training Value = "1_Training"

Testing partition size: 35 Label: Testing Value = "2_Testing"

Validation partition size: 30 Label: Validation Value = "3_Validation"

Total size: 100%

Values: Use system-defined values ("1", "2" and "3")

Append labels to system-defined values

Use labels as values

Repeatable partition assignment

Seed: 8245561 Generate

Use unique field to assign partitions:

OK Cancel Apply Reset

Analysis node

Analysis of [MacCohort_kr]

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	337	80.05%	184	76.99%
Wrong	84	19.95%	55	23.01%
Total	421		239	

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	72	23	1	1	0	0
CONTROL	0	207	0	0	0	0
PSP	2	16	4	0	0	0
bvFTD	6	7	2	28	1	0
nfPPAunspc	7	7	0	6	3	1
svPPA	3	0	0	1	0	23

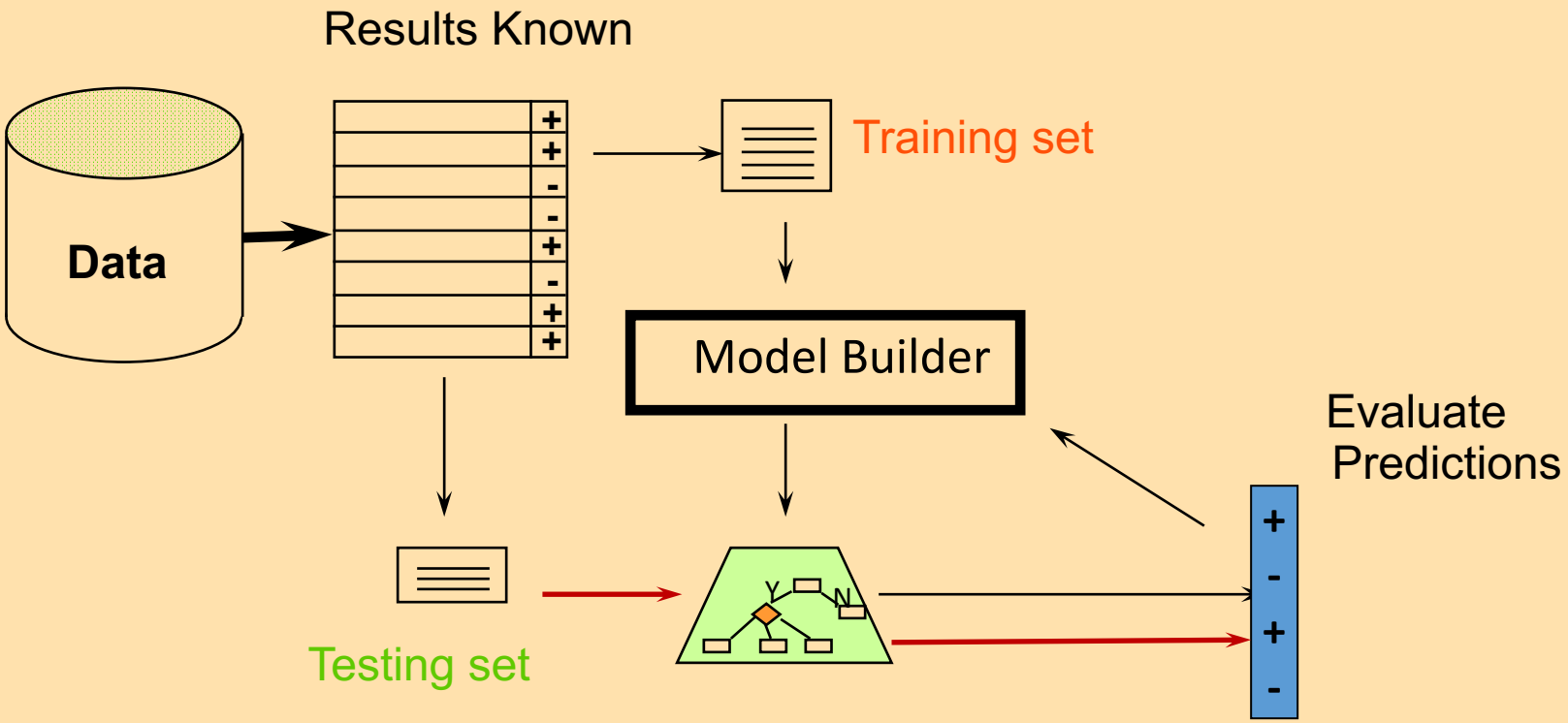
'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	36	15	1	0	0	2
CONTROL	2	106	0	0	0	0
PSP	3	14	4	1	0	0
bvFTD	1	4	2	17	0	0
nfPPAunspc	1	7	1	0	5	0
svPPA	1	0	0	0	0	16

OK

Parameter estimation

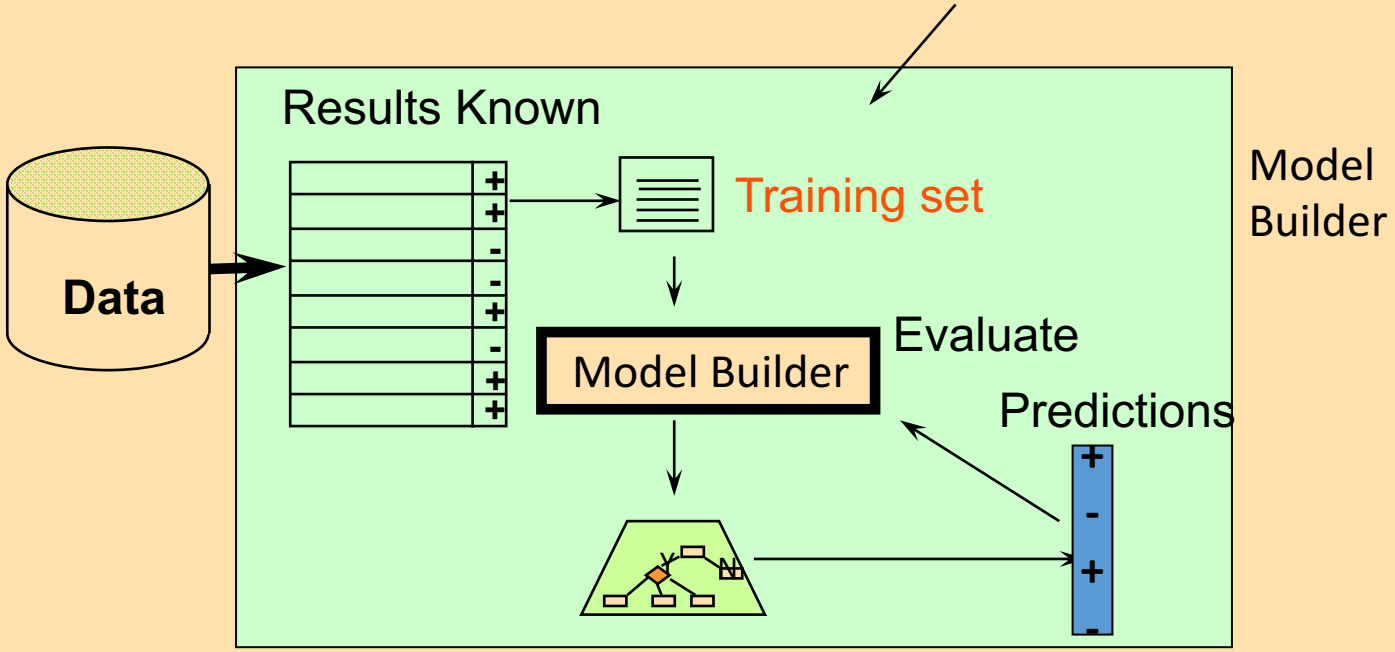
- Many classifiers (and regression approaches) have inbuilt "tuning parameters" (like choosing k for nearest neighbors)
- Some learning schemes operate in two stages:
  - Stage 1: builds the basic structure
  - Stage 2: optimizes parameter settings
  - Note that instead of 2 stages for the KNN example, we did all model fitting and optimization in the training set, which risks overfitting for the best choice of k
- The test data can't be used for parameter tuning!
- Proper procedure uses three sets: **training data, validation data, and test data**
  - Validation data is used to optimize parameters
- Once evaluation is complete, *all the data* can be used to build the final classifier

Training-test



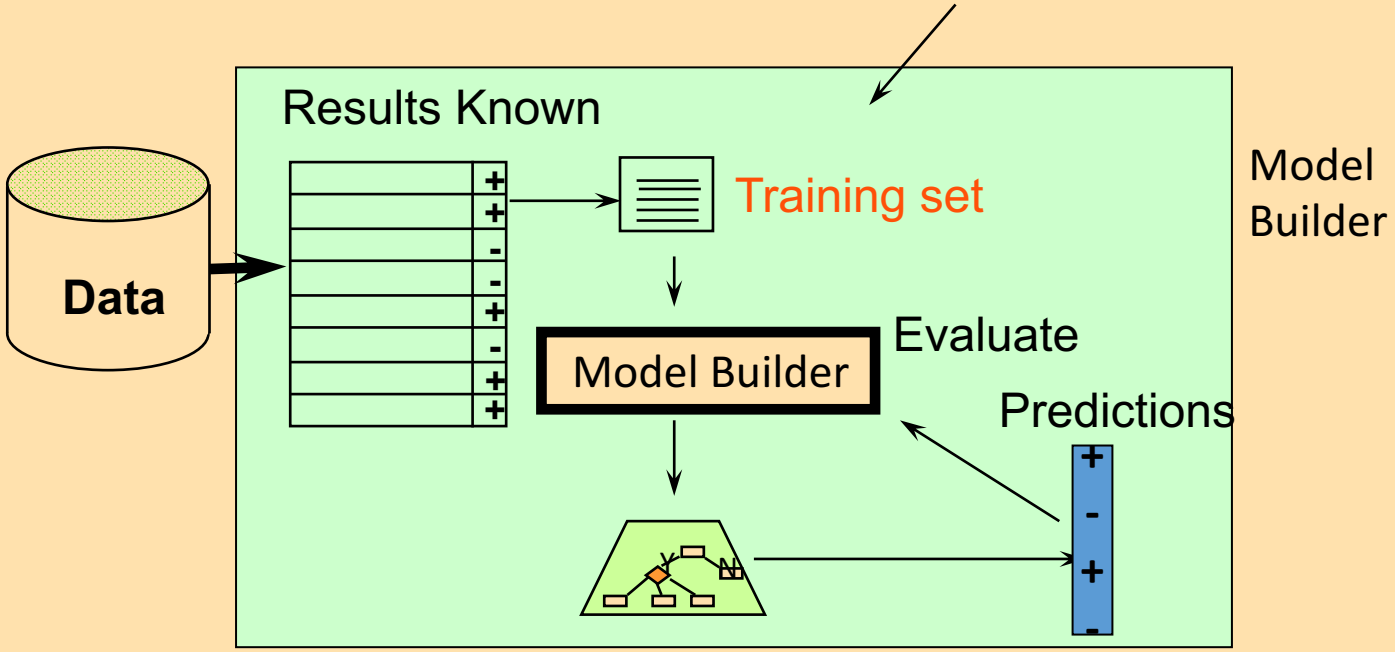
Train-validate-test

Do this green box for multiple values of the procedures parameter(s), e.g. K = 3, 4, 5, 6 for K-nearest neighbors



Train-validate-test

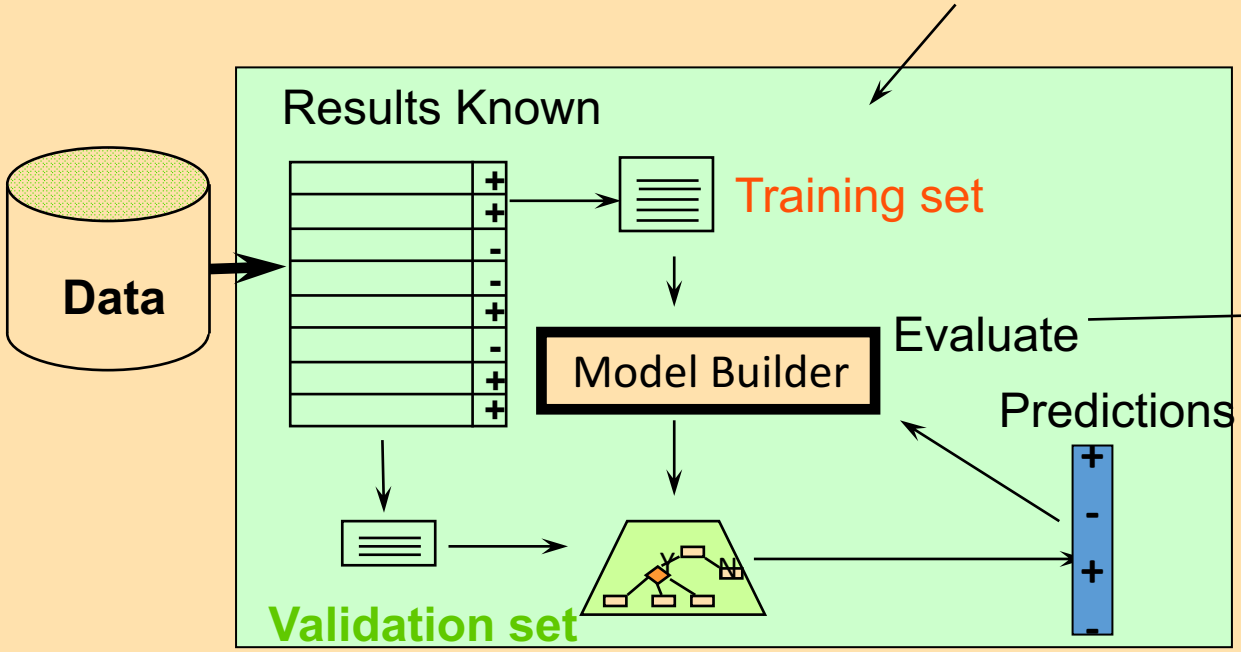
Do this green box for multiple values of the procedures parameter(s), e.g. K = 3, 4, 5, 6 for K-nearest neighbors



Each model based on training set has potentially been over-fitted… so utilize a validation set to determine best fitting model

Train-validate-test

Do this green box for multiple values of the procedures parameter(s), e.g. K = 3, 4, 5, 6 for K-nearest neighbors

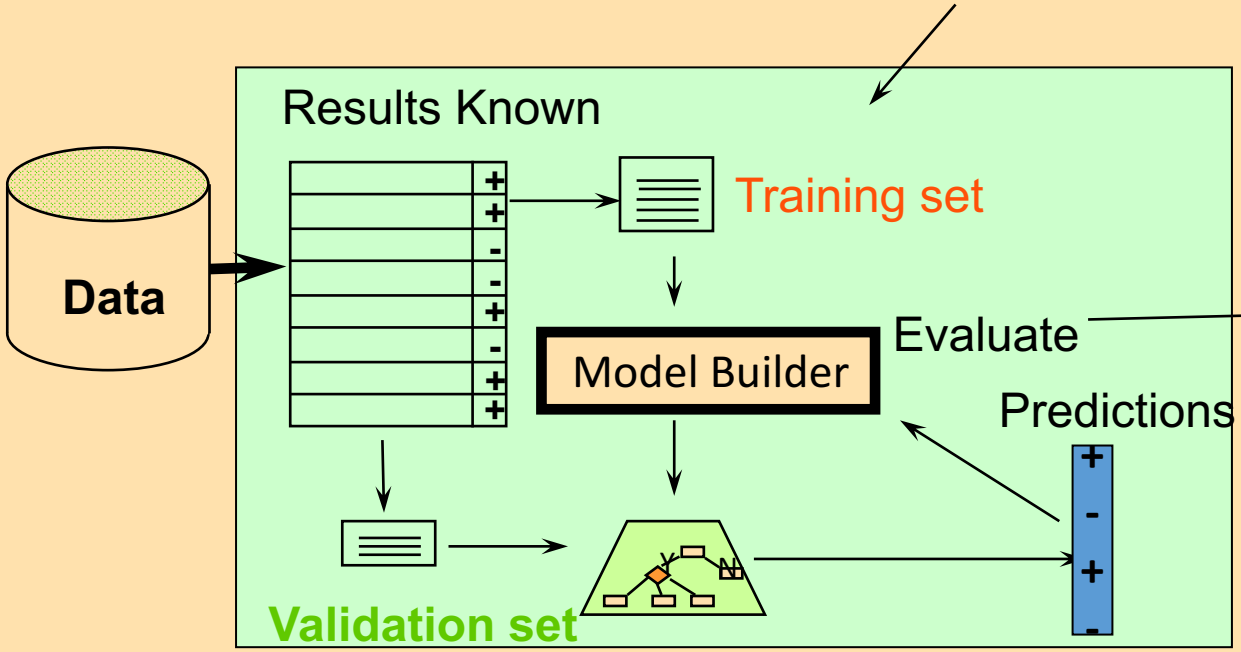


Model Builder

Choose best parameter values based on **validation set** results – evaluate model for each parameter value and choose the best

Train-validate-test

Do this green box for multiple values of the procedures parameter(s), e.g. K = 3, 4, 5, 6 for K-nearest neighbors

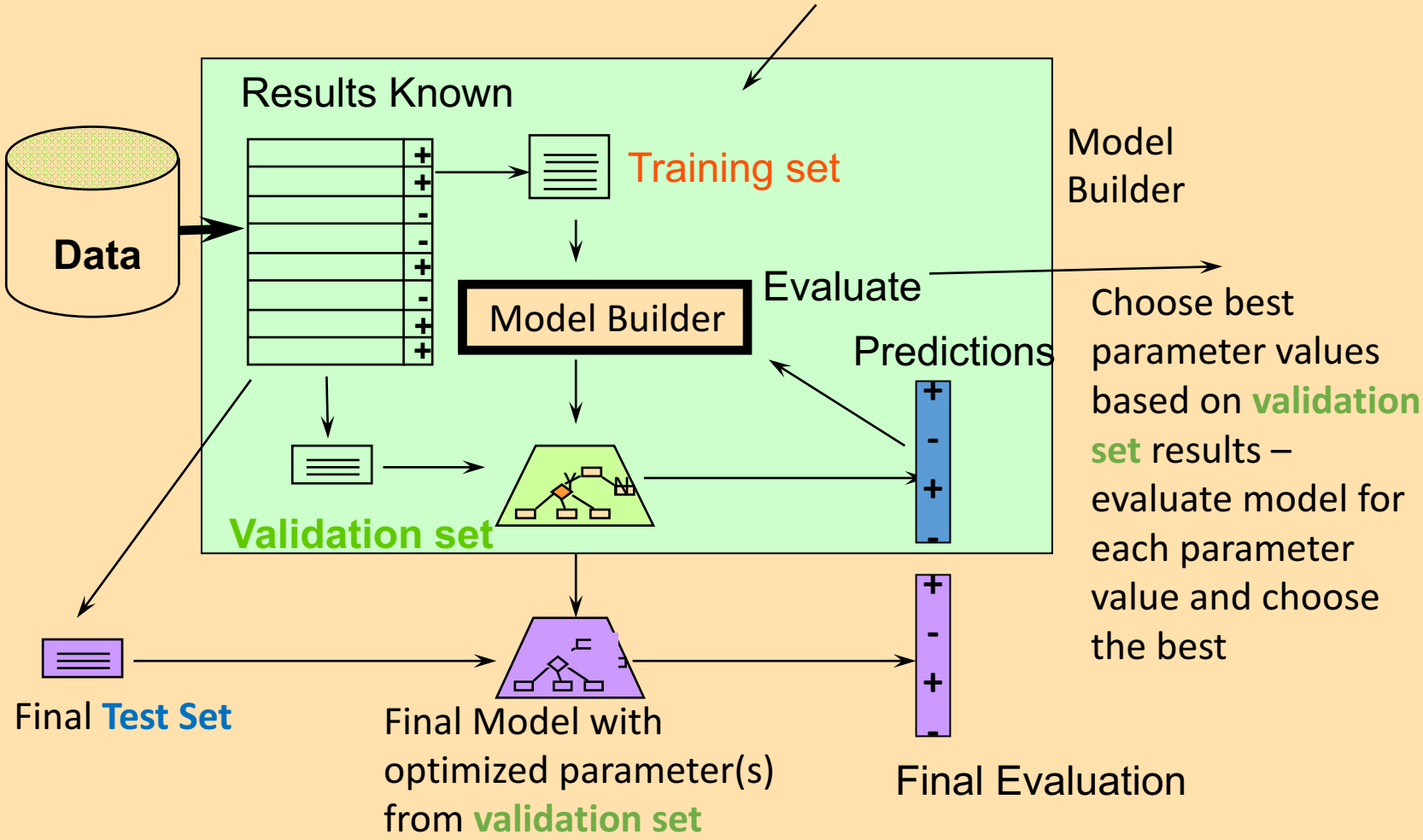


Choose best parameter values based on validation set results – evaluate model for each parameter value and choose the best

But because we are choosing from many models we have again potentially over-fitted… so utilize a test set to determine accuracy of best fitting model

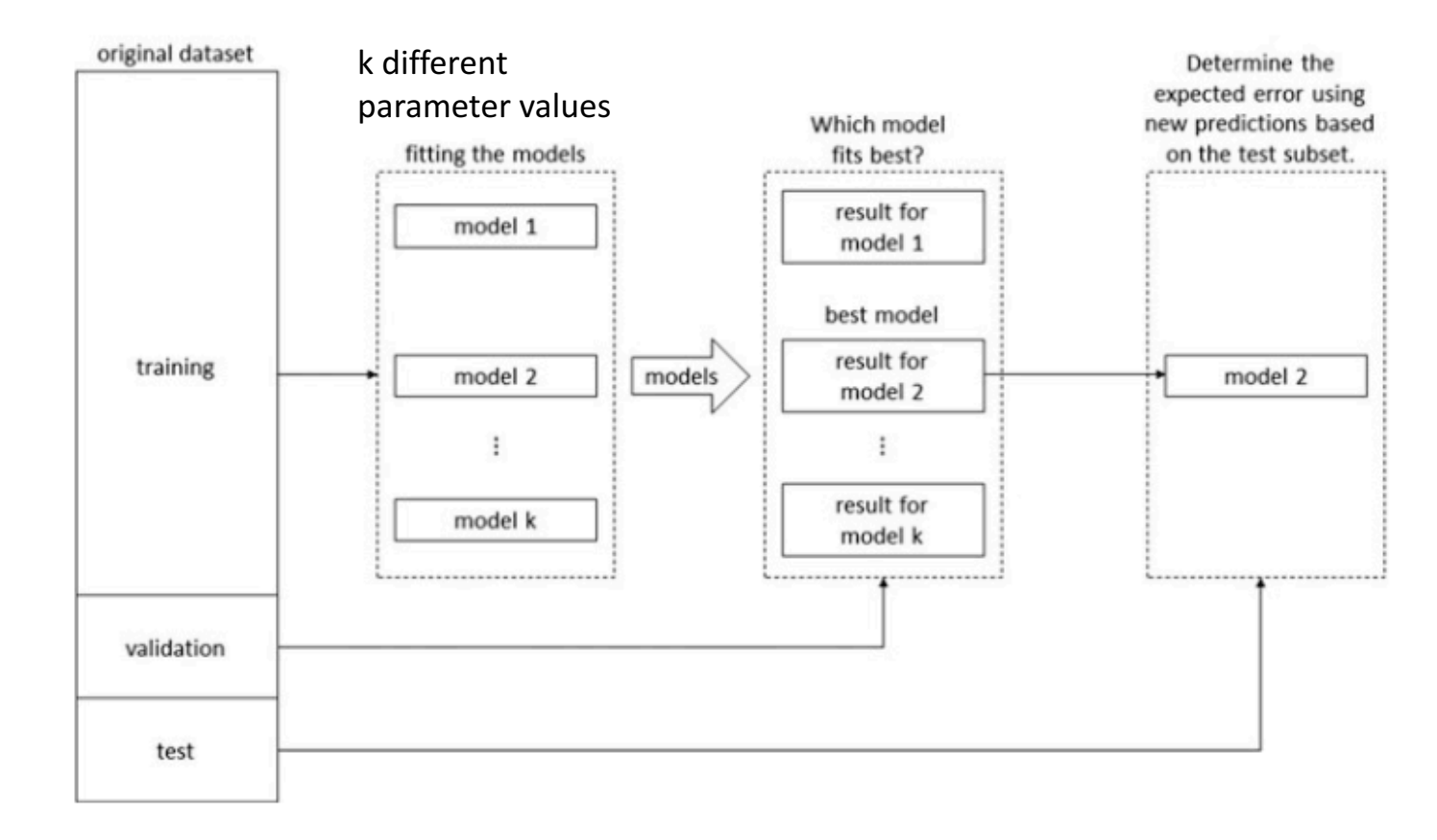
Train-validate-test

Do this green box for multiple values of the procedures parameter(s), e.g. K = 3, 4, 5, 6 for K-nearest neighbors



Train-validate-test

A different illustration from IBM Modeler text book (Wendler and Gröttrup)



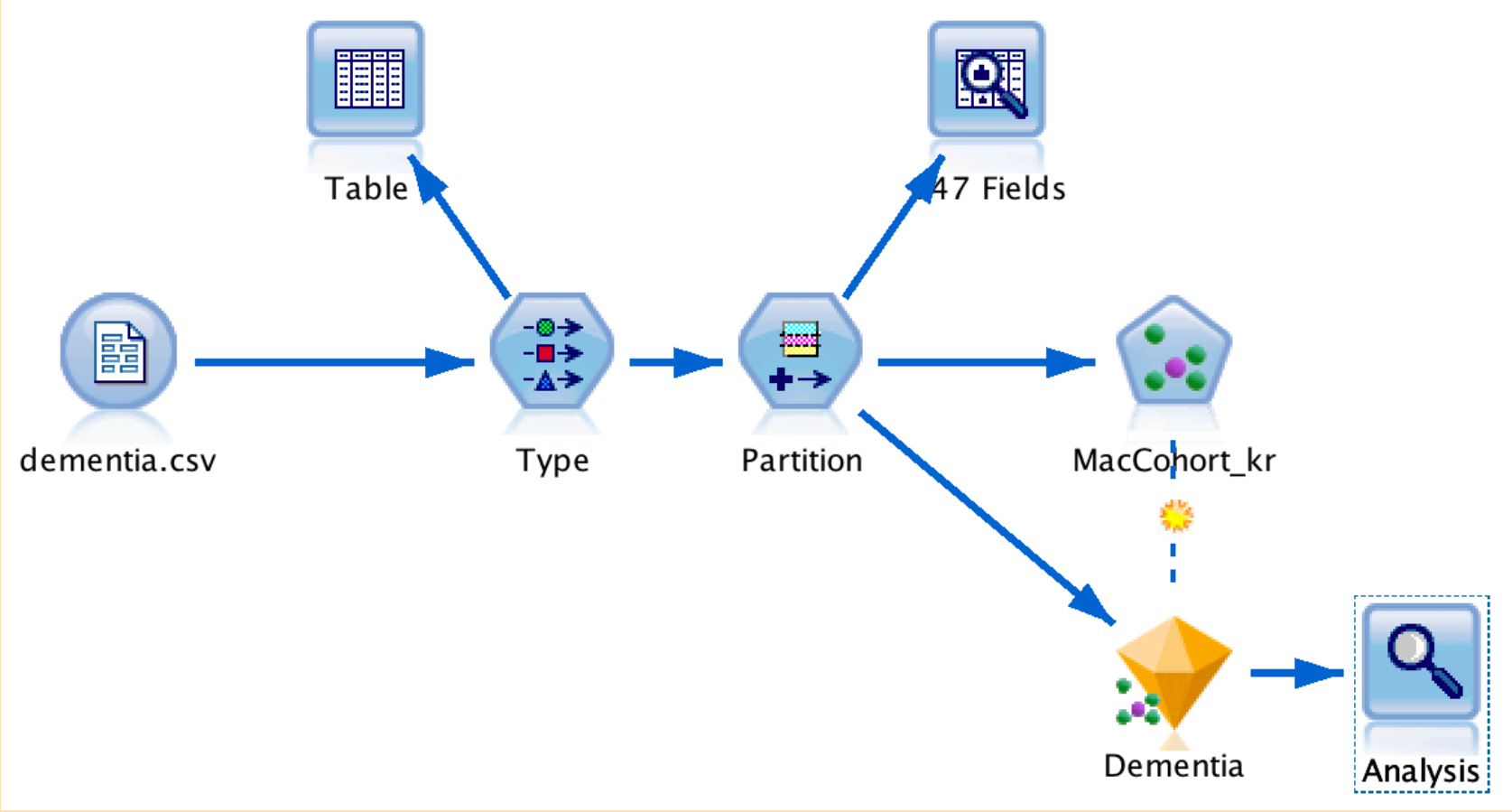
Warning: IBM Modeler goes against convention here:

IBM = training/test/validate
convention = training/validate/test

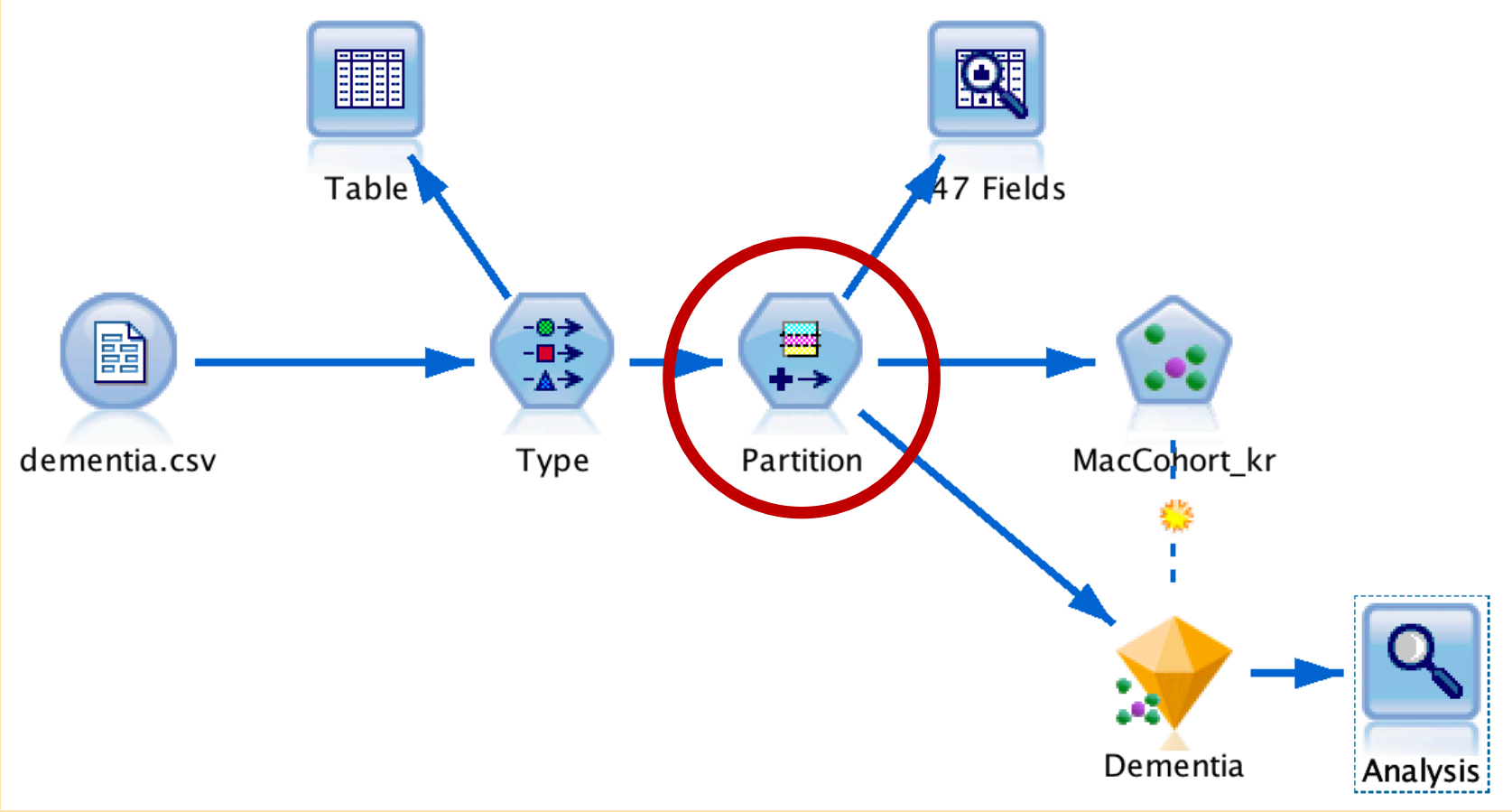
We will use convention in the slides, but be careful to switch test and validate definitions when working in Modeler

Let's see this applied to the dementia dataset using K-nearest neighbors

Modeler: K-Nearest Neighbors



Modeler: K-Nearest Neighbors



Revised Partition Node

Partition

Generate Preview

Settings Annotations

Partition field: Partition

Partitions: Train and test Train, test and validation

Training partition size:	50	Label:	Training	Value =	"1_Training"
Testing partition size:	25	Label:	Testing	Value =	"2_Testing"
Validation partition size:	25	Label:	Validation	Value =	"3_Validation"

Total size: 100%

Values: Use system-defined values ("1", "2" and "3")

Append labels to system-defined values

Use labels as values

Repeatable partition assignment

Seed: 1227565 Generate

Use unique field to assign partitions:

OK Cancel Apply Reset

Analysis node

Best model was still for k = 6, so didn't make much of a difference in this case

Analysis of [MacCohort_kr] #1

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing		3_Validation	
Correct	270	81.08%	124	76.54%	126	76.36%
Wrong	63	18.92%	38	23.46%	39	23.64%
Total	333		162		165	

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	53	18	1	1	0	0
CONTROL	1	173	0	0	0	0
PSP	3	13	4	0	0	0
bvFTD	2	7	1	16	1	0
nfPPAunspc	4	5	0	4	3	0
svPPA	2	0	0	0	0	21

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	26	12	1	0	0	1
CONTROL	0	70	0	0	0	0
PSP	1	6	1	1	0	0
bvFTD	2	3	0	18	0	0
nfPPAunspc	2	5	0	1	1	1
svPPA	1	0	0	1	0	8

'Partition' = 3_Validation	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	29	8	0	0	0	1
CONTROL	1	70	0	0	0	0
PSP	1	12	2	0	0	0
bvFTD	2	2	3	11	0	0
nfPPAunspc	2	4	1	1	4	0
svPPA	1	0	0	0	0	10

OK

Analysis node

Best model was still for k = 6, so didn't make much of a difference in this case

Analysis of [MacCohort_kr] #1

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing		3_Validation	
Correct	270	81.08%	124	76.54%	126	76.36%
Wrong	63	18.92%	38	23.46%	39	23.64%
Total	333		162		165	

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	53	18	1	1	0	0
CONTROL	1	173	0	0	0	0
PSP	3	13	4	0	0	0
bvFTD	2	7	1	16	1	0
nfPPAunspc	4	5	0	4	3	0
svPPA	2	0	0	0	0	21

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	26	12	1	0	0	1
CONTROL	0	70	0	0	0	0
PSP	1	6	1	1	0	0
bvFTD	2	3	0	18	0	0
nfPPAunspc	2	5	0	1	1	1
svPPA	1	0	0	1	0	8

'Partition' = 3_Validation	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	29	8	0	0	0	1
CONTROL	1	70	0	0	0	0
PSP	1	12	2	0	0	0
bvFTD	2	2	3	11	0	0
nfPPAunspc	2	4	1	1	4	0
svPPA	1	0	0	0	0	10

OK

Cross-validation

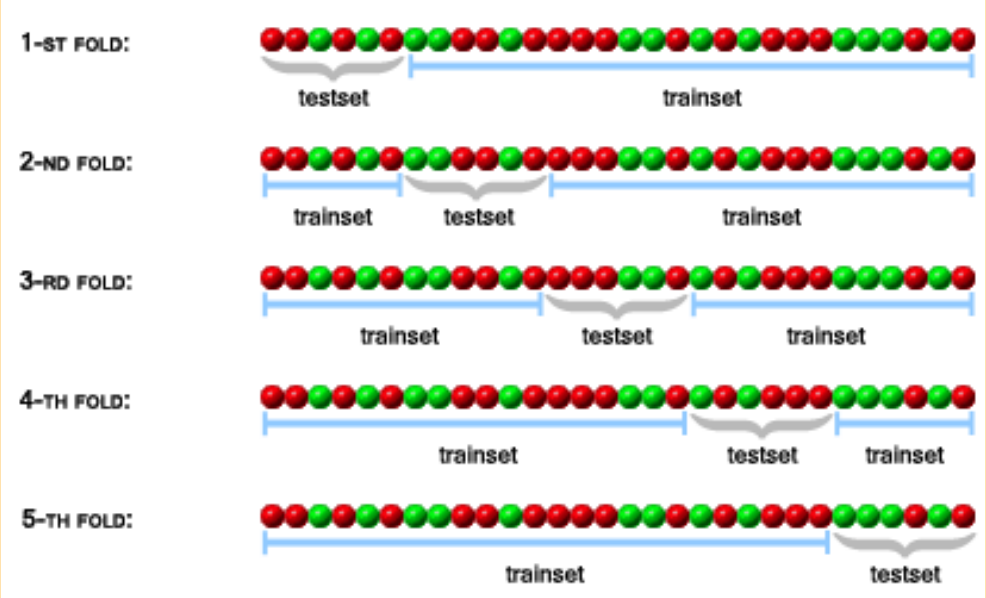
What if you don’t have enough data to split 3 or even 2 ways? If data is expensive, it is a big sacrifice to reserve a large proportion of data for validation and testing

Cross-validation

- *Cross-validation*
  - First step: data is split into *k* subsets of equal size
  - Second step: each subset is used in turn for validation and the remainder for training
- This is called *k-fold cross-validation* (10-fold is common choice, also 5-fold for smaller datasets approx. <200 and even *N-fold* for very small datasets, called leave-one-out)
- Often the subsets are stratified before the cross-validation is performed – e.g. same proportion of each diagnostic class in each fold
- The error estimates are averaged across folds to yield an overall error estimate

E.g. 5-fold cross-validation

- Break up data into 5 groups of the same size
- Hold aside one group for testing and use the rest to build model
- Repeat

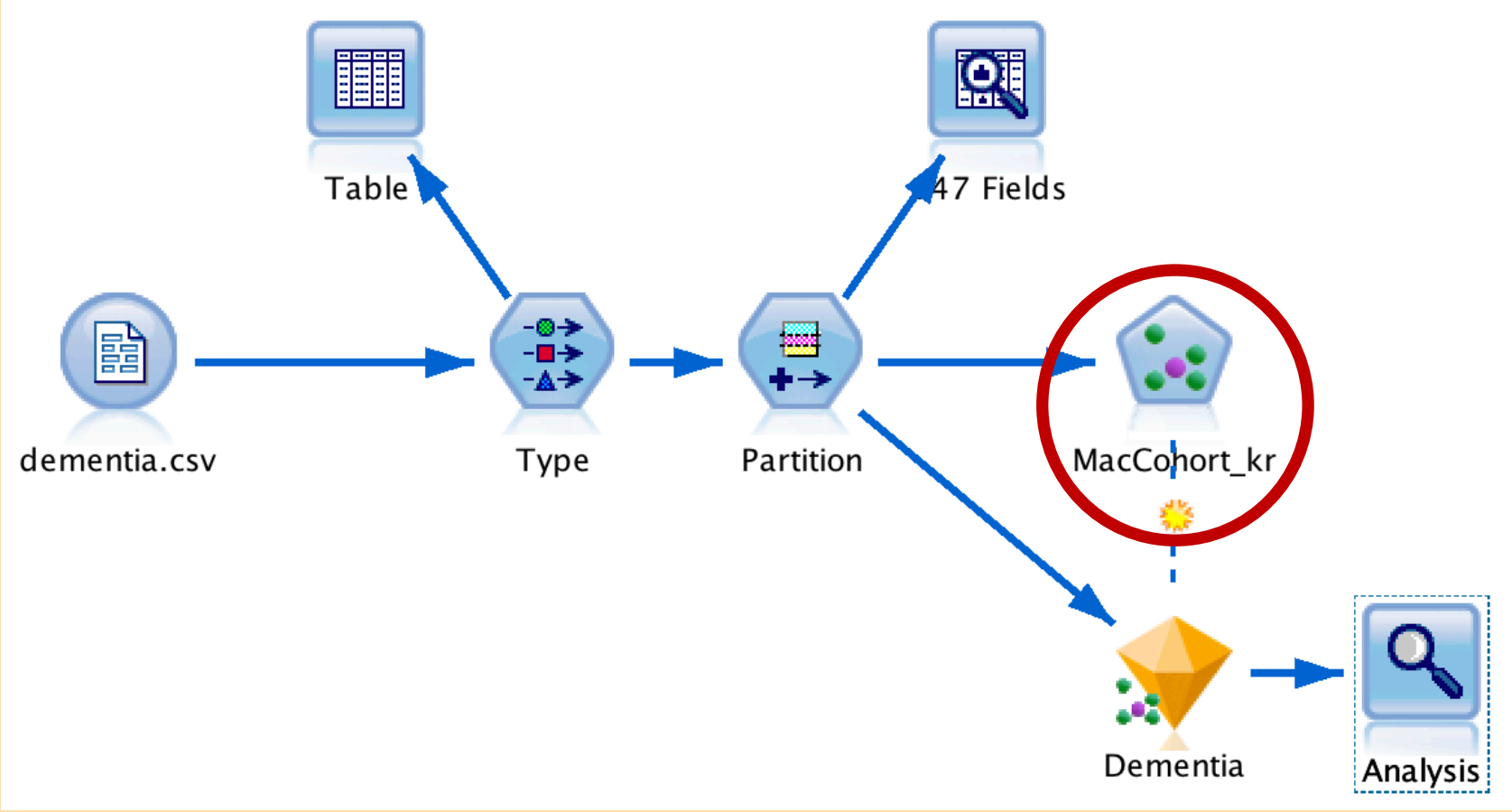


- When every group has had a turn at being the test set then you can estimate the classification accuracy based on the combined Test data (i.e. the 5 testsets)

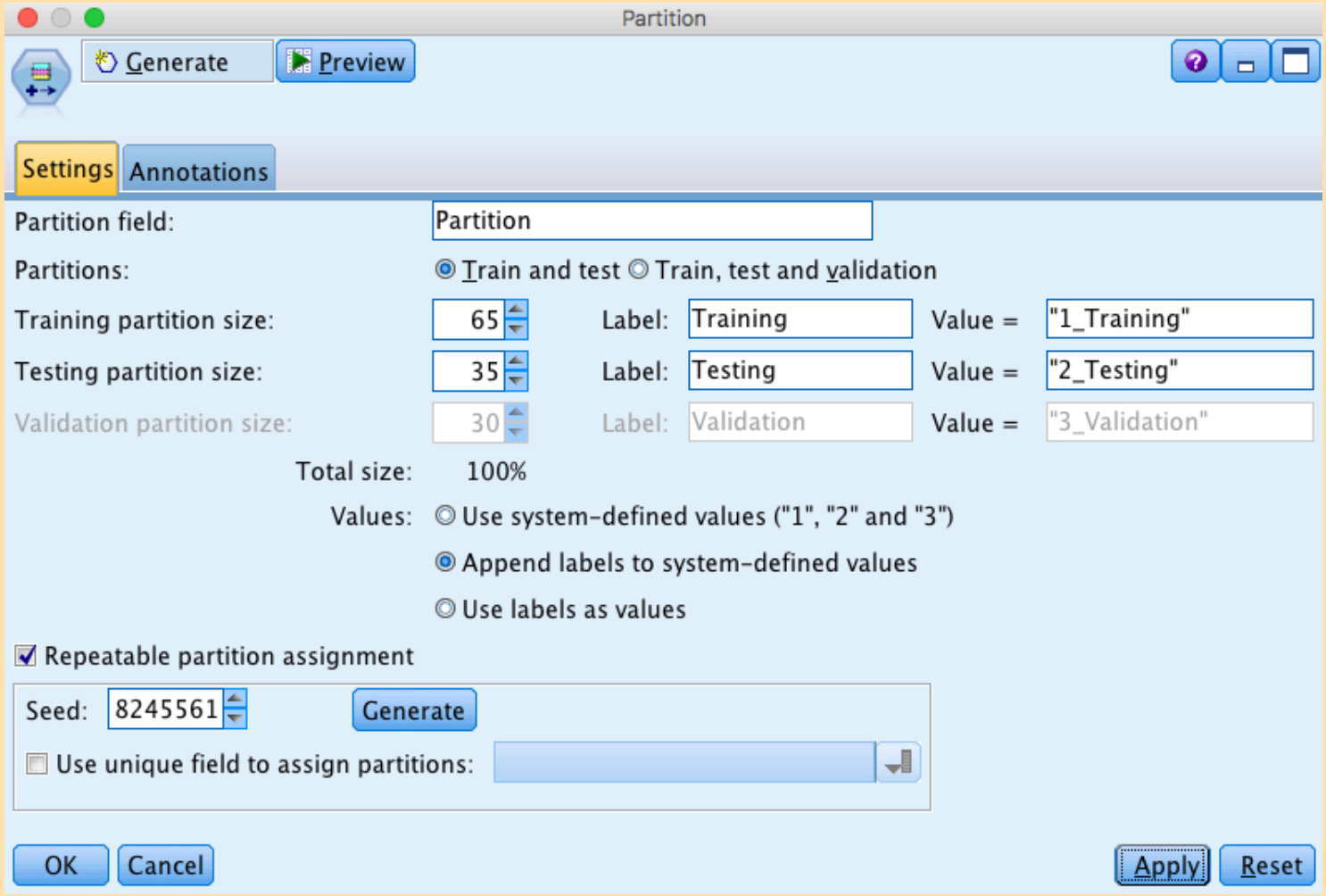
Cross-validation

- Notice that at no time is the classification accuracy based on data that was used to fit the model – so there is no over-fitting
- And all of the data gets used in the process unlike Training-Test
- As a final step you can then rebuild the classifier using all the data for predicting future observations – this model will (on average) be better than any of the individual models in the cross-validation because it uses more data
- Cross-validation can be performed in 1) the validation step (i.e. for parameter estimation – easy for some algorithms in Modeler), or it can be performed in 2) the test step (i.e. for determining prediction quality – more difficult in Modeler)
- Let's look at cross-validation for the value of K in K-nearest neighbors (validation step)

Modeler: K-Nearest Neighbors

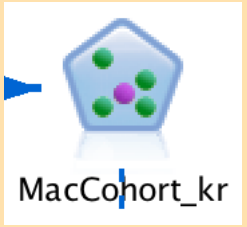


Partition Node



We are going to cross-validate within the Training set so don't need a separate validation set

KNN Node



MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

☐ Perform feature selection

Forward selection is used to evaluate features for inclusion. To force a feature into the model prior to forward selection, select it in the Forced entry list.

Forced entry:

Stopping Criterion

◉ Stop when the specified number of features have been selected

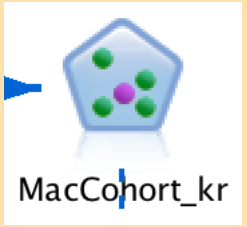
Number to select: 10

◎ Stop when the change in the absolute error ratio is less than or equal to the minimum

Minimum change: 0.010

OK | Run | Cancel | Apply | Reset

KNN Node



MacCohort_kr

Objectives | Fields | Settings | Annotations

- Model
- Neighbors
- Feature Selection
- Cross-Validation
- Analyze

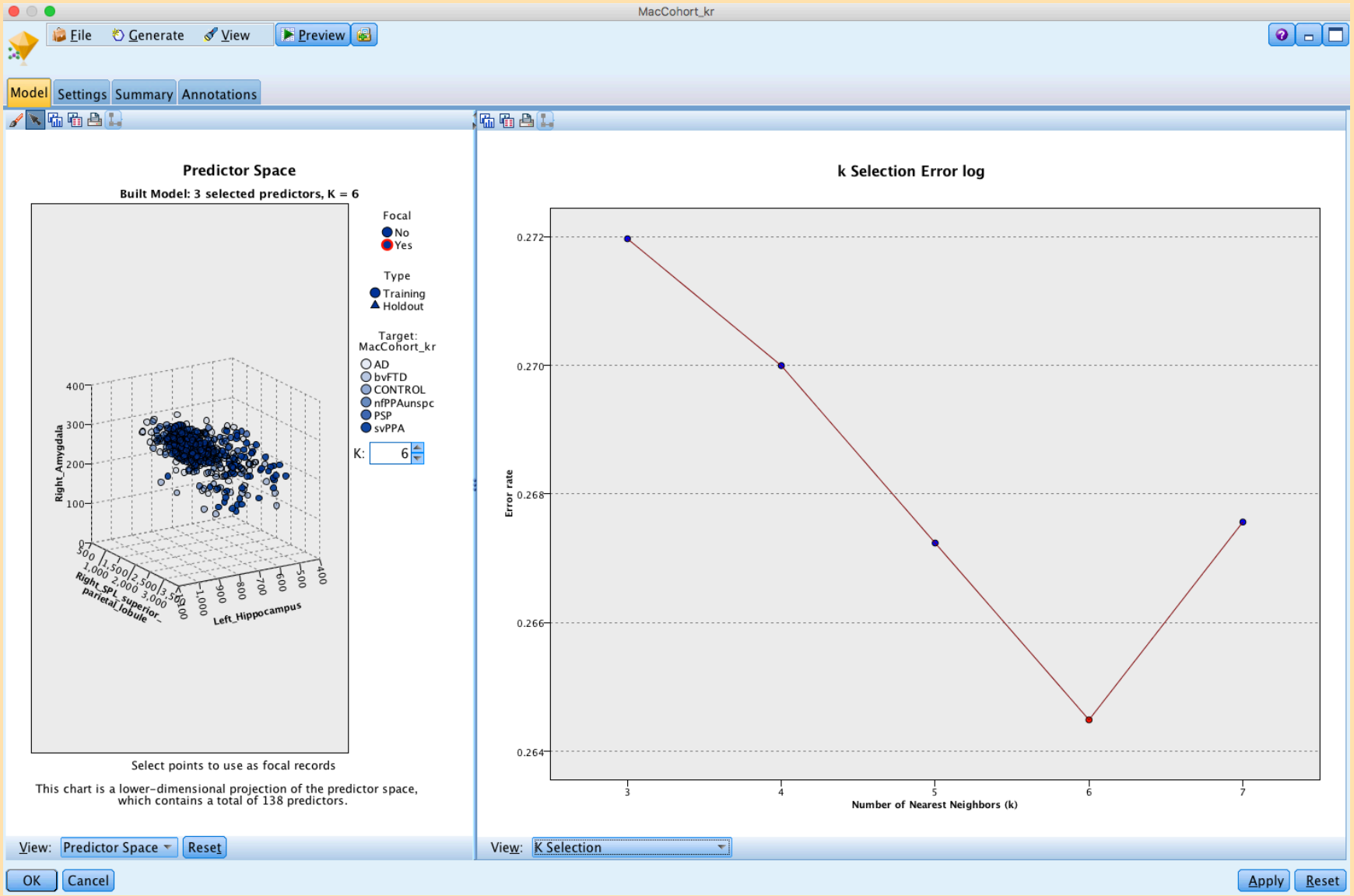
Cross-Validation Folds

V-fold cross-validation is performed if you choose automatic k selection but do not choose feature selection.

- (•) Randomly assign cases to folds
  - Number of folds: 10
  - [x] Set random seed — Seed: 12345
- ( ) Use field to assign cases

OK | Run | Cancel | Apply | Reset

Golden nugget



Analysis node

Again, best model was still for k = 6, so didn't make much of a difference in this case

Analysis of [MacCohort_kr]

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $KNN-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	330	78.38%	182	76.15%
Wrong	91	21.62%	57	23.85%
Total	421		239	

Coincidence Matrix for $KNN-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	69	25	0	3	0	0
CONTROL	2	205	0	0	0	0
PSP	2	19	1	0	0	0
bvFTD	2	11	0	29	0	2
nfPPAunspc	8	6	2	5	3	0
svPPA	3	0	0	1	0	23

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	34	19	0	1	0	0
CONTROL	0	107	1	0	0	0
PSP	0	18	2	2	0	0
bvFTD	1	4	0	18	1	0
nfPPAunspc	0	9	0	0	5	0
svPPA	0	1	0	0	0	16

OK

Another Warning:

IBM text book (Wendler and Gröttrup) wrongly considers cross-validation to be a simple *train-test-validate* analysis

This flies in the face of the definition for cross-validation!

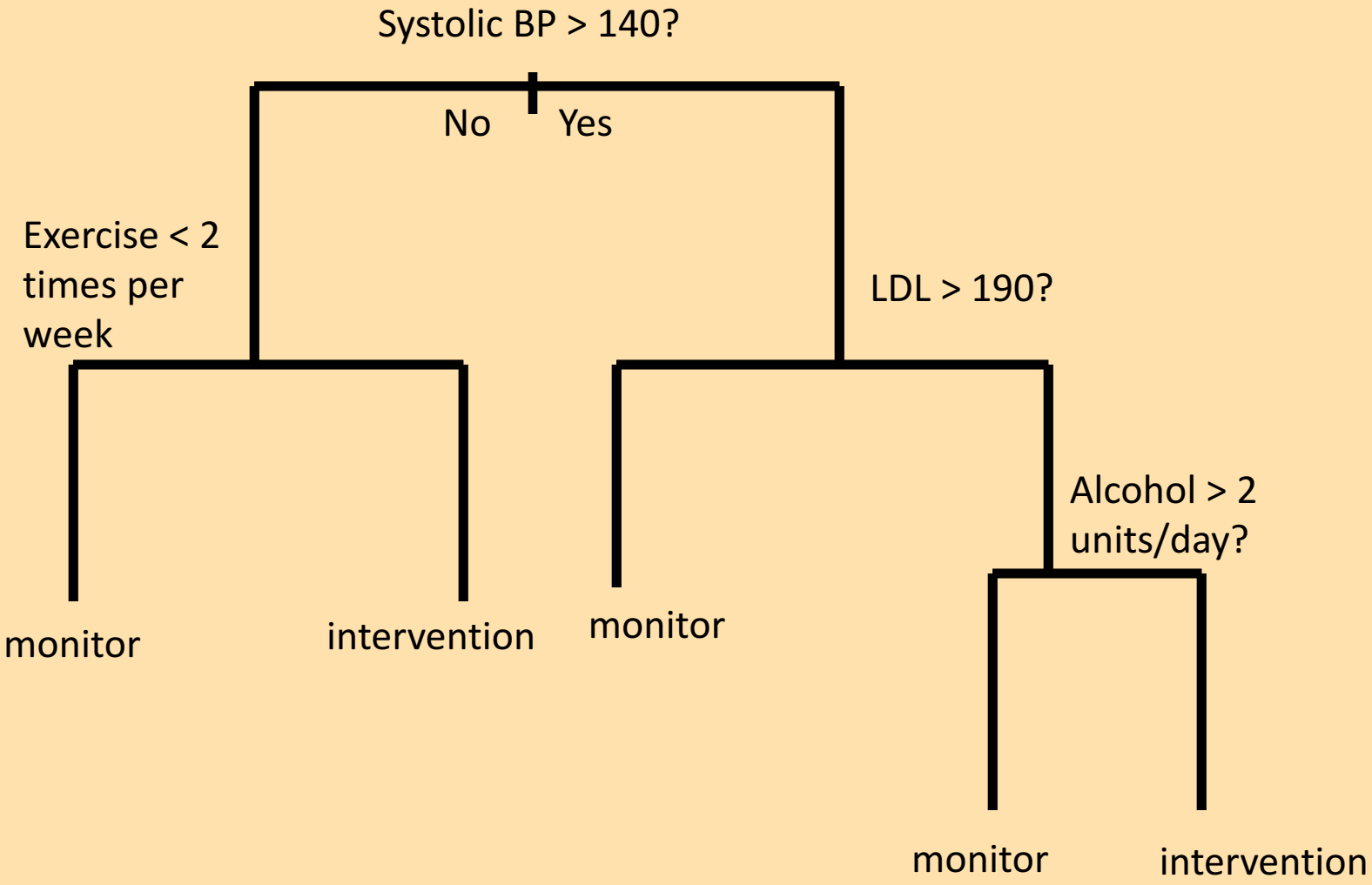
IBM Modeler gets it right though

Recursive Partitioning (Decision Trees)

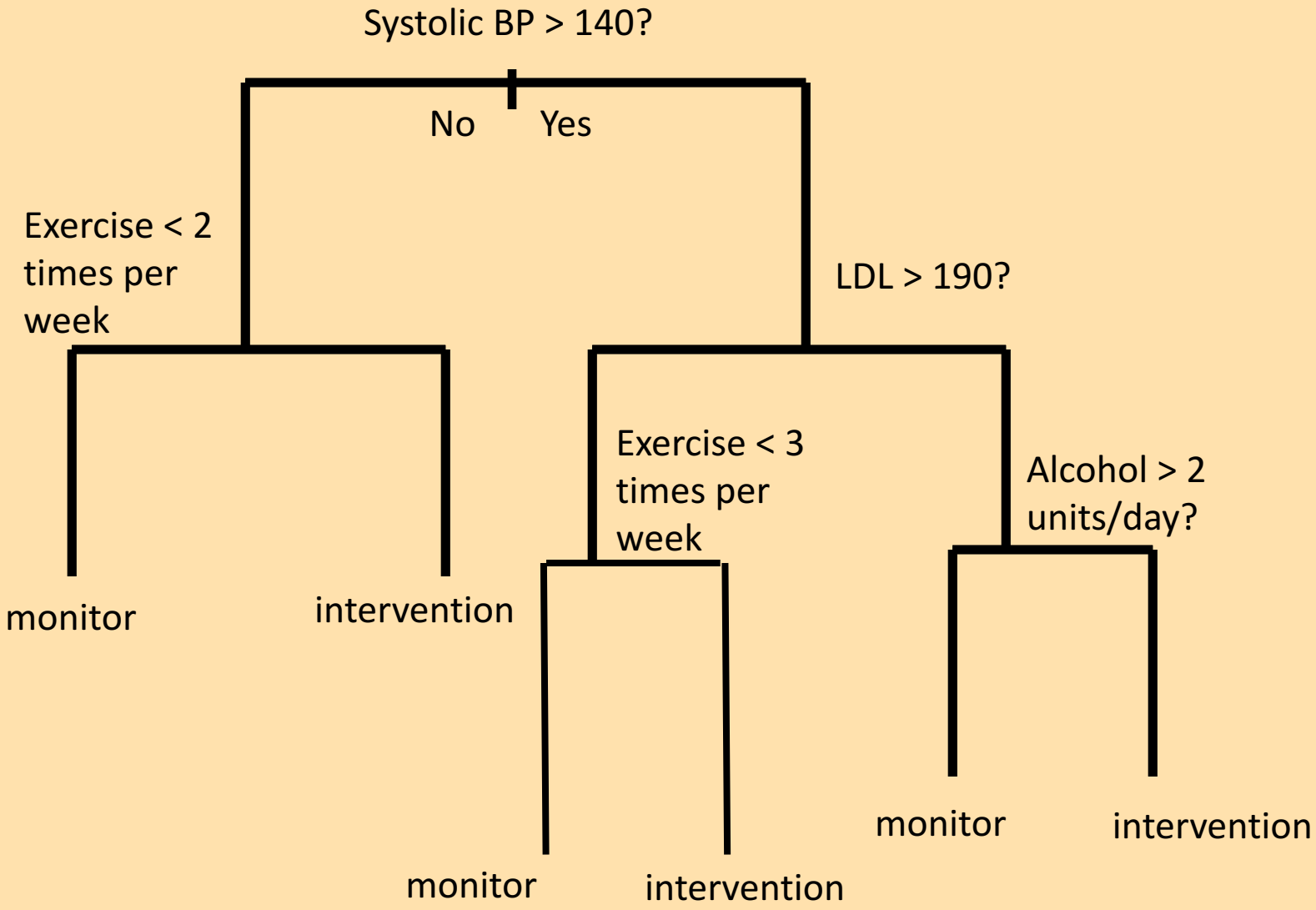
Recursive Partitioning

- Classification based on a series of binary decision cut-points e.g.,
  - Is systolic blood pressure greater that 140?
  - If yes, is LDL Cholesterol greater than 190?
  - If no, exercise > 2 times per week
- At the end of the series of questions a *decision node* is reached (note that the series of questions may not be the same for every subject – it depends which branch of the tree you go down)

Risk of heart problem



Risk of heart problem



Growing Trees

- Greedy algorithmic approach
- Start with first best split on a single variable (e.g. based on *Gini Index**)
  - The *Gini Index* is related to classification error, but penalizes errors more as the proportion of observations in a node is further from 0.5 – favors *pure nodes*
  - A *pure node* is a decision node where all of the observed data is of a single class
- Then for each branch look for the best split on a single variable for the observations within that branch
- Keep going until each branch is pure or has less than some number of observations (e.g. 5)

*Some algorithms, such as C5.0, use different criteria

Modeler decision tree methods

Decision tree	Method	SPSS Modeler node
CART	– Gini coefficient – Twoing criteria	C&R Tree
C5.0	Information Gain (Entropy)	C5.0
CHAID	Chi-squared statistics	CHAID
QUEST	Significance statistics	QUEST

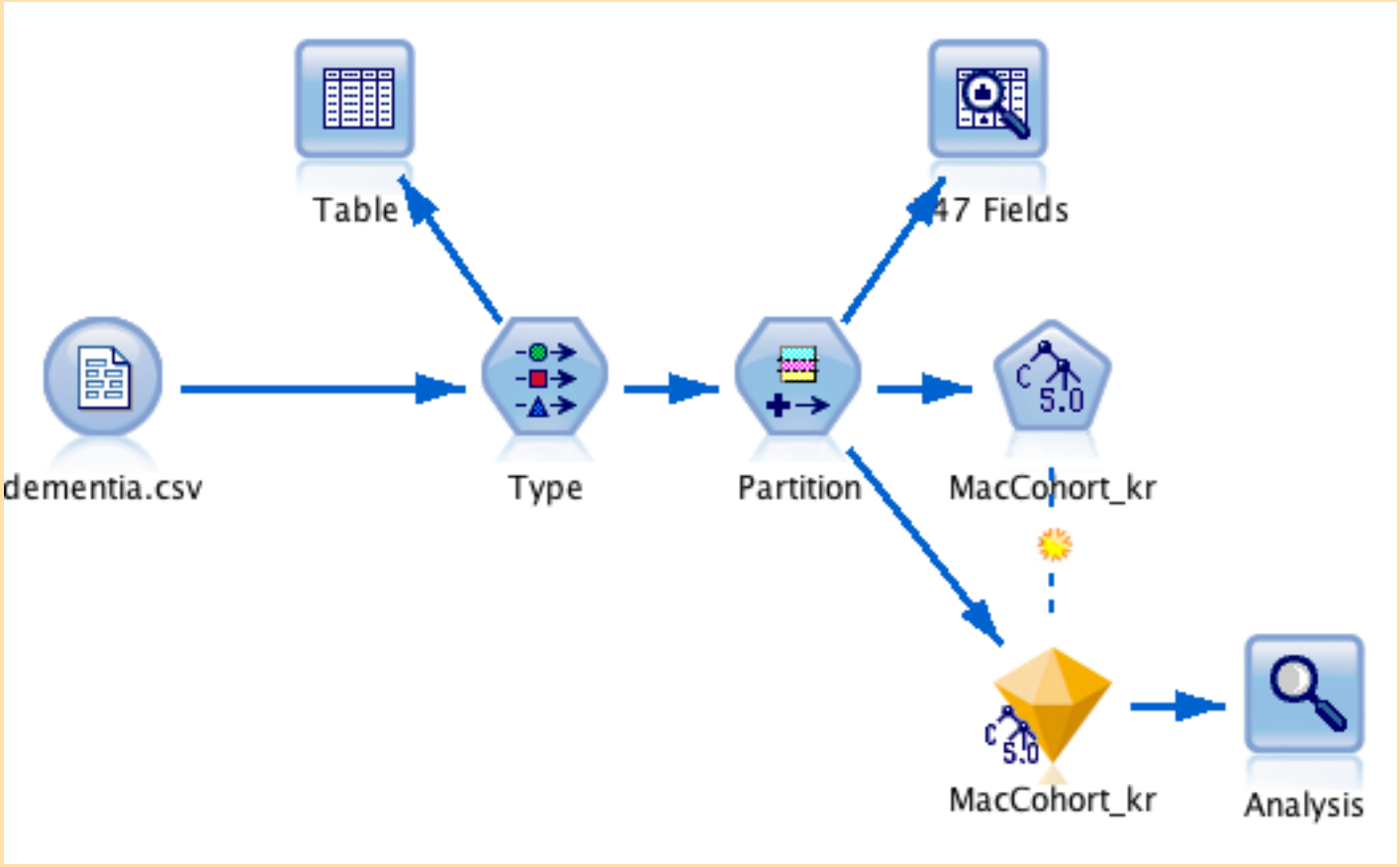
Pruning Trees

- Full tree after growing is typically too complex
- Goal is to find the sub-tree (contained in the fully grown tree) that minimizes a cost-complexity criterion.
- The cost-complexity criterion is a trade-off between the classification error rate and the complexity (size) of the tree
- The best balance (weighting) for the trade-off is determined via cross-validation to minimize the classification error rate

Recursive Partitioning

- Recursive partitioning methods are a very popular classification approach and come in many flavors: Classification and Regression Trees (CART), C5.0, and cutting-edge *ensemble* extensions like Random Forests (average over a bunch of trees – hence forest)
- Partitioning on cut-points approach is very popular in medicine, because mimics decision approach of clinicians

C5.0 Example Stream



Partition Node

Partition

Generate Preview

Settings Annotations

Partition field: Partition

Partitions: Train and test Train, test and validation

Training partition size:	70	Label:	Training	Value =	"1_Training"
Testing partition size:	30	Label:	Testing	Value =	"2_Testing"
Validation partition size:	25	Label:	Validation	Value =	"3_Validation"

Total size: 100%

Values: Use system-defined values ("1", "2" and "3")

Append labels to system-defined values

Use labels as values

Repeatable partition assignment

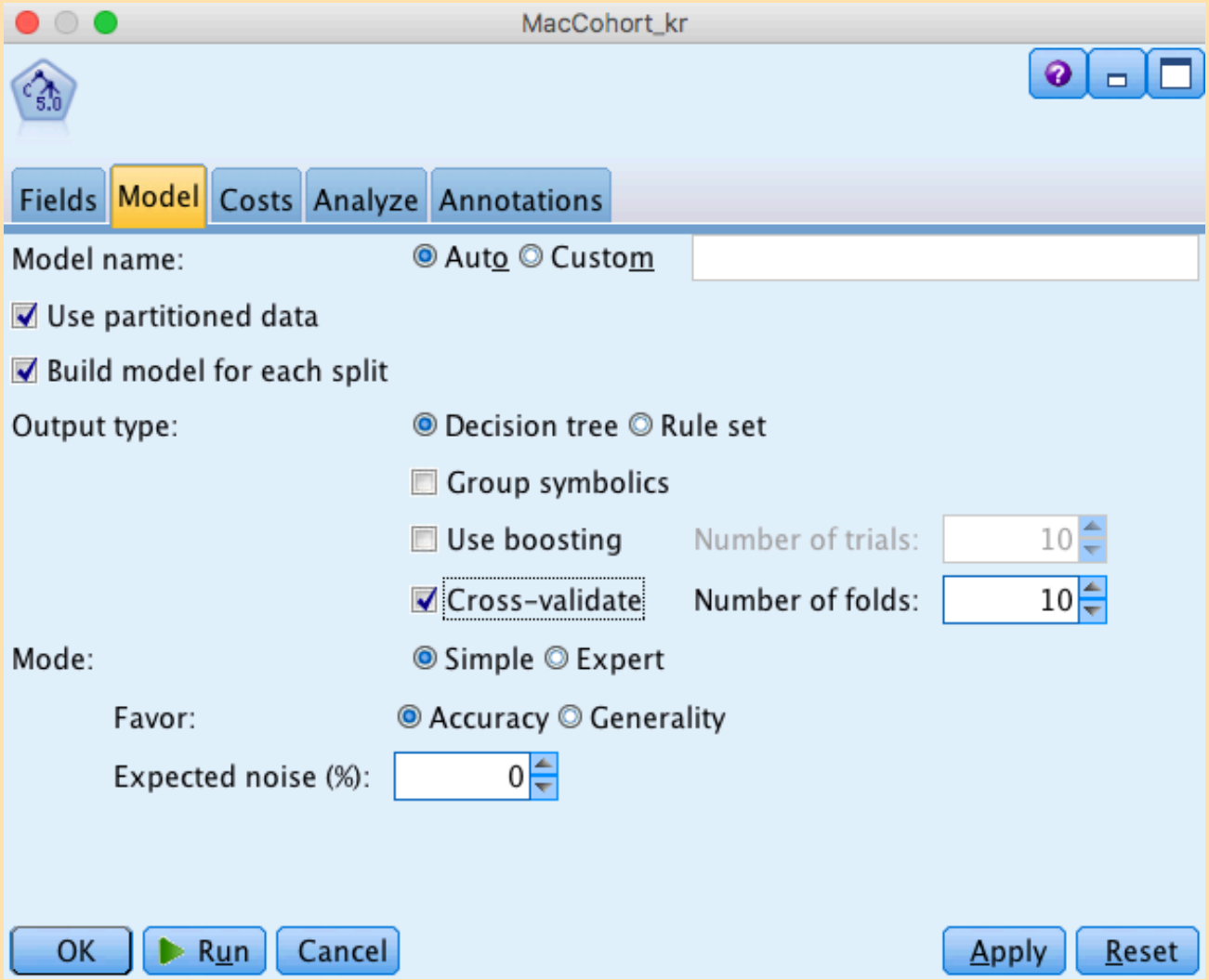
Seed: 896468 Generate

Use unique field to assign partitions:

OK Cancel Apply Reset

C5.0 Node


5.0
MacCohort_kr


MacCohort_kr
Fields
Model
Costs
Analyze
Annotations
Model name:
Auto
Custom
Use partitioned data
Build model for each split
Output type:
Decision tree
Rule set
Group symbolics
Use boosting
Number of trials:
10
Cross-validate
Number of folds:
10
Mode:
Simple
Expert
Favor:
Accuracy
Generality
Expected noise (%):
0
OK
Run
Cancel
Apply
Reset

C5.0 Node

MacCohort_kr

Fields | Model | Costs | Analyze | Annotations

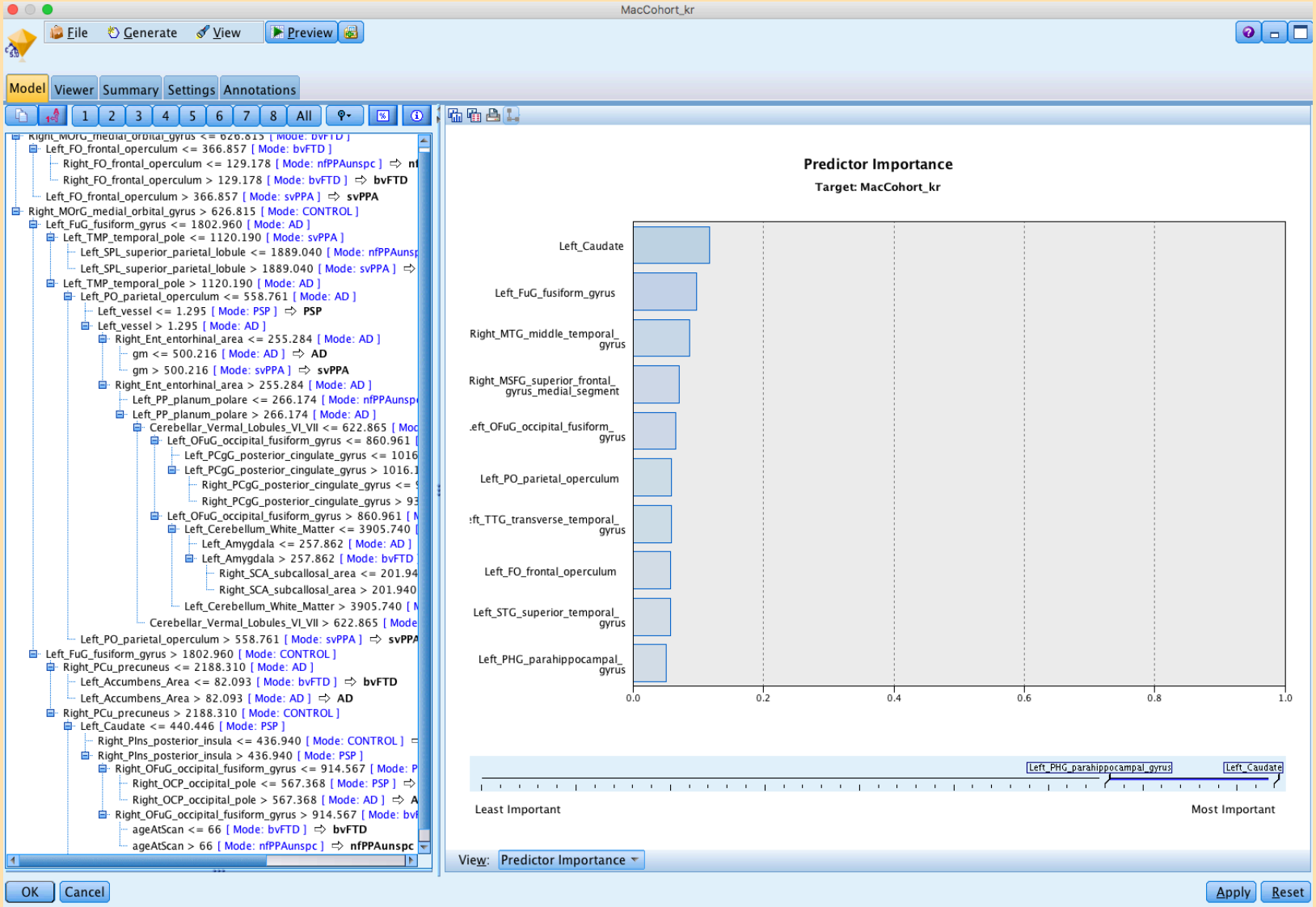
☐ Use misclassification costs

Predicted

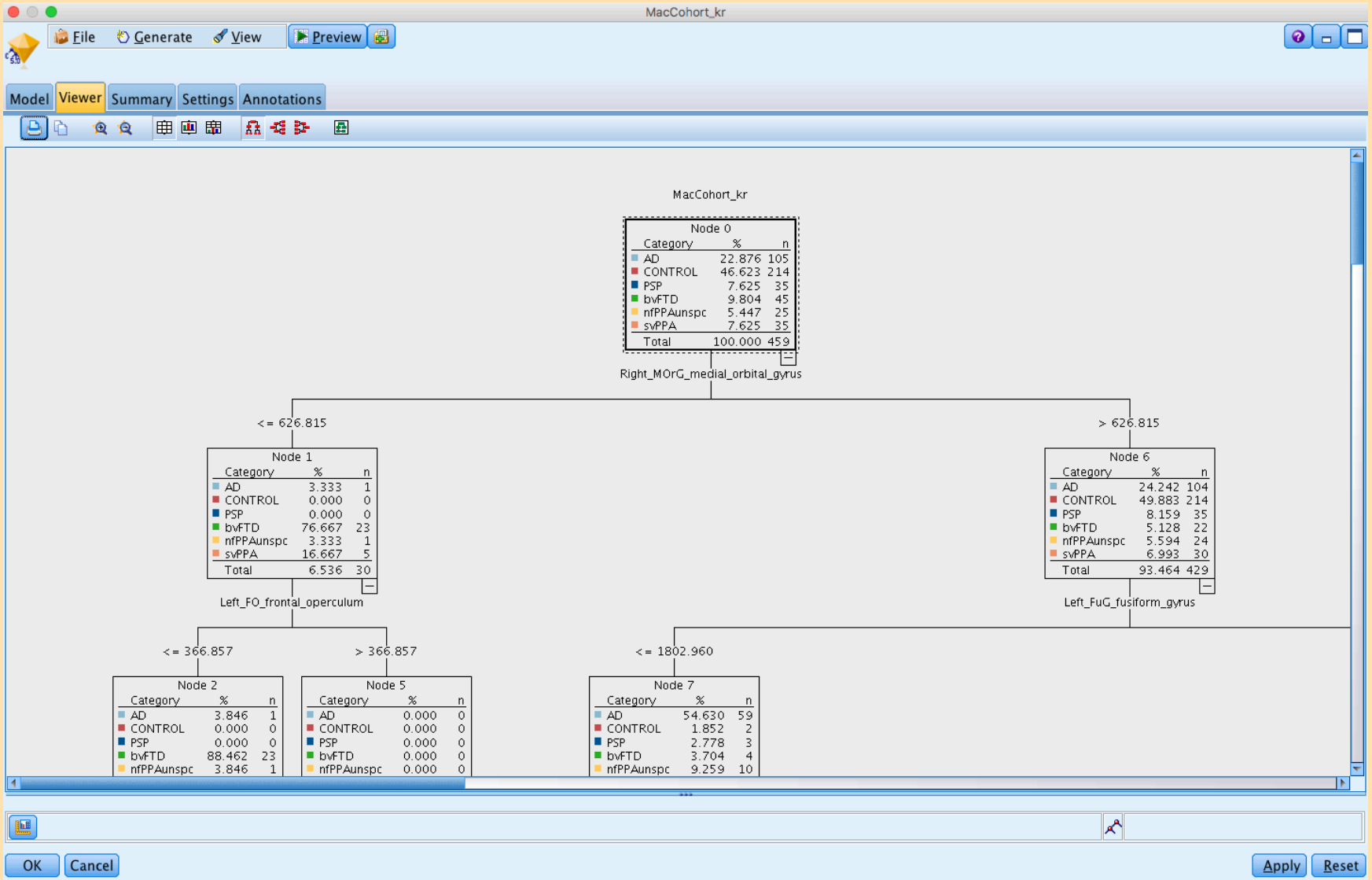
Actual	AD	CONTROL	PSP	bvFTD	nfPPAu...	svPPA
AD	0.0	1.0	1.0	1.0	1.0	1.0
CONTROL	1.0	0.0	1.0	1.0	1.0	1.0
PSP	1.0	1.0	0.0	1.0	1.0	1.0
bvFTD	1.0	1.0	1.0	0.0	1.0	1.0
nfPPAunspc	1.0	1.0	1.0	1.0	0.0	1.0
svPPA	1.0	1.0	1.0	1.0	1.0	0.0

OK | Run | Cancel | Apply | Reset

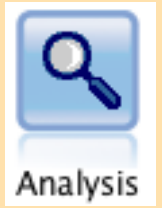
Golden Nugget



Golden Nugget



Analysis Node



Analysis of [MacCohort_kr]

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $C-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	424	92.37%	129	64.18%
Wrong	35	7.63%	72	35.82%
Total	459		201	

Coincidence Matrix for $C-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	98	4	1	1	1	0
CONTROL	2	210	1	0	1	0
PSP	1	4	27	2	1	0
bvFTD	1	6	2	34	2	0
nfPPAunspc	0	2	0	0	22	1
svPPA	0	1	0	0	1	33

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA
AD	26	6	4	5	3	2
CONTROL	7	79	8	4	2	1
PSP	2	4	1	1	1	0
bvFTD	1	2	4	16	0	0
nfPPAunspc	1	2	6	4	0	0
svPPA	1	0	0	0	1	7

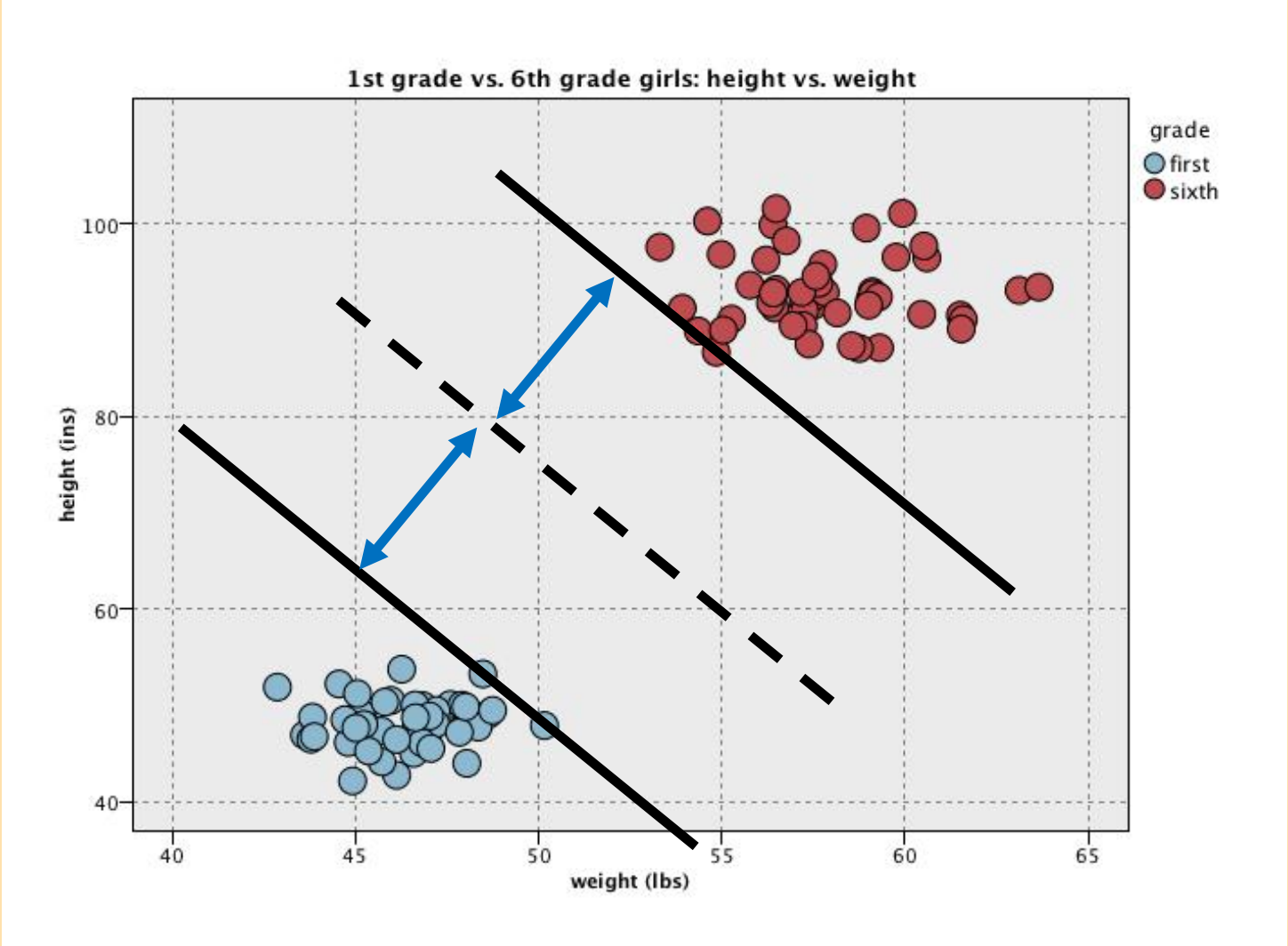
OK

Support vector machines

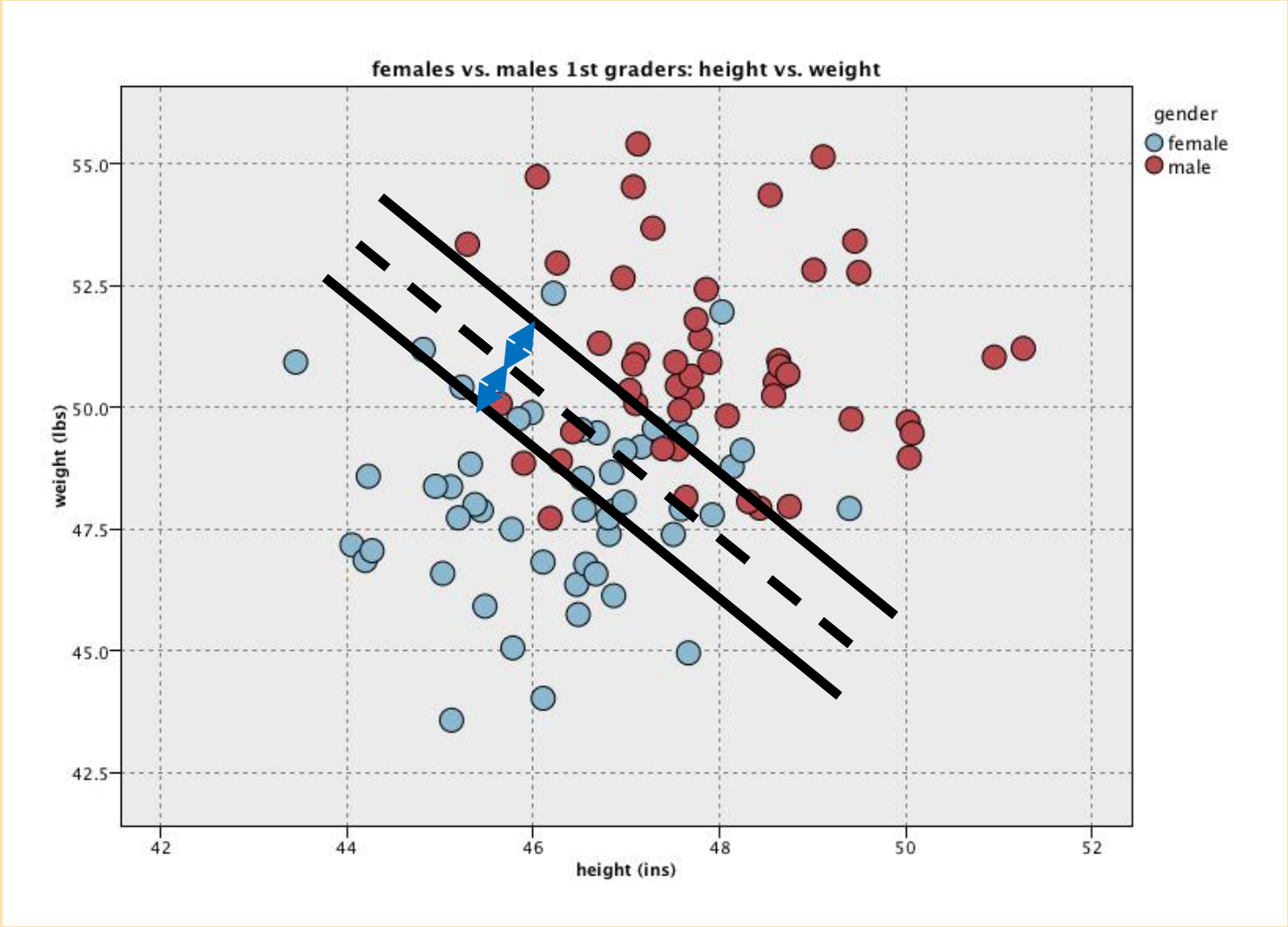
Support vector machines

- Idea is to generate a (non-linear) hyper-plane that optimally separates points from different classes
- Developed in computer science community in 1990s

Maximizing margins

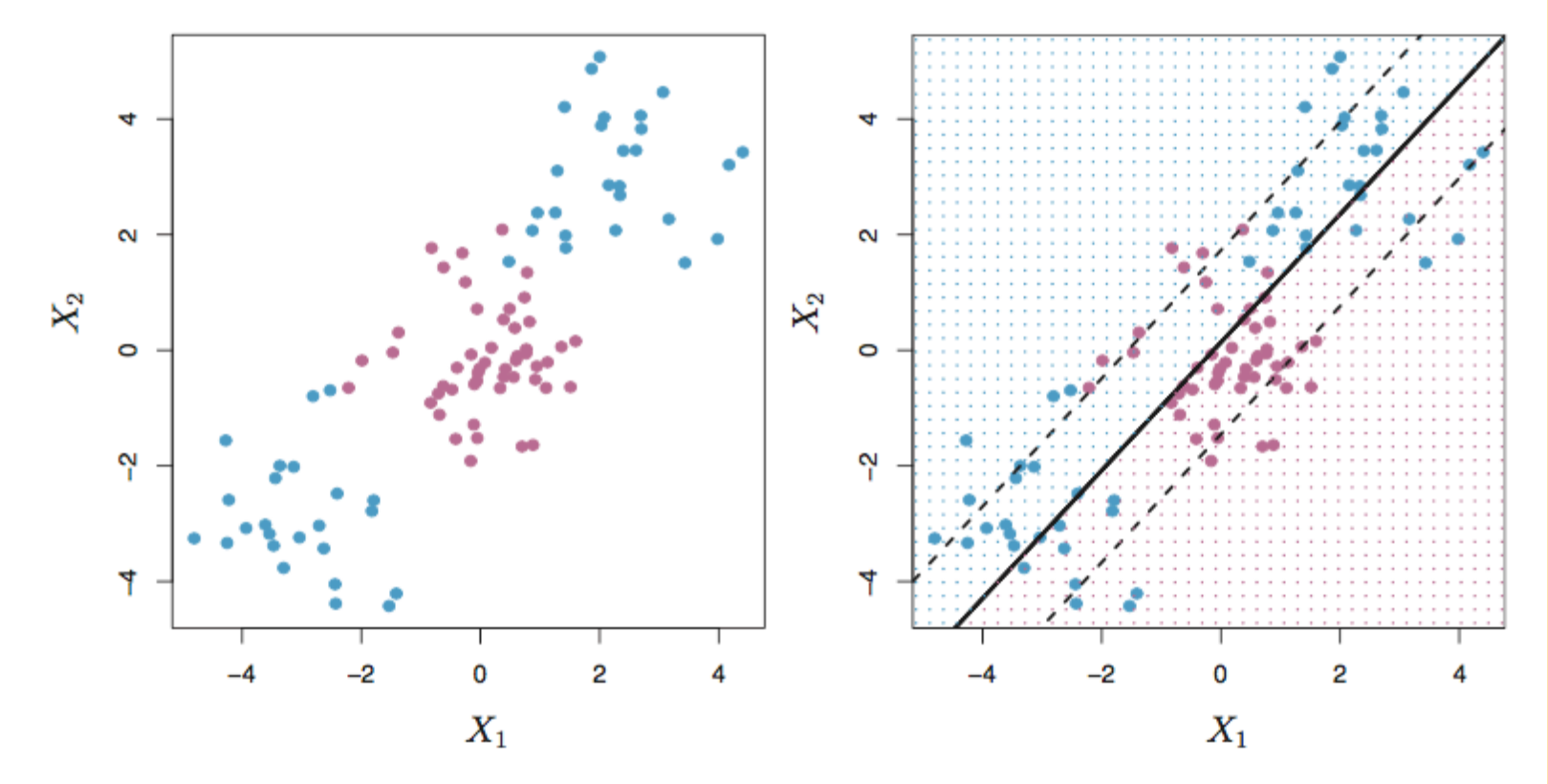


However, usually we have overlapping data…

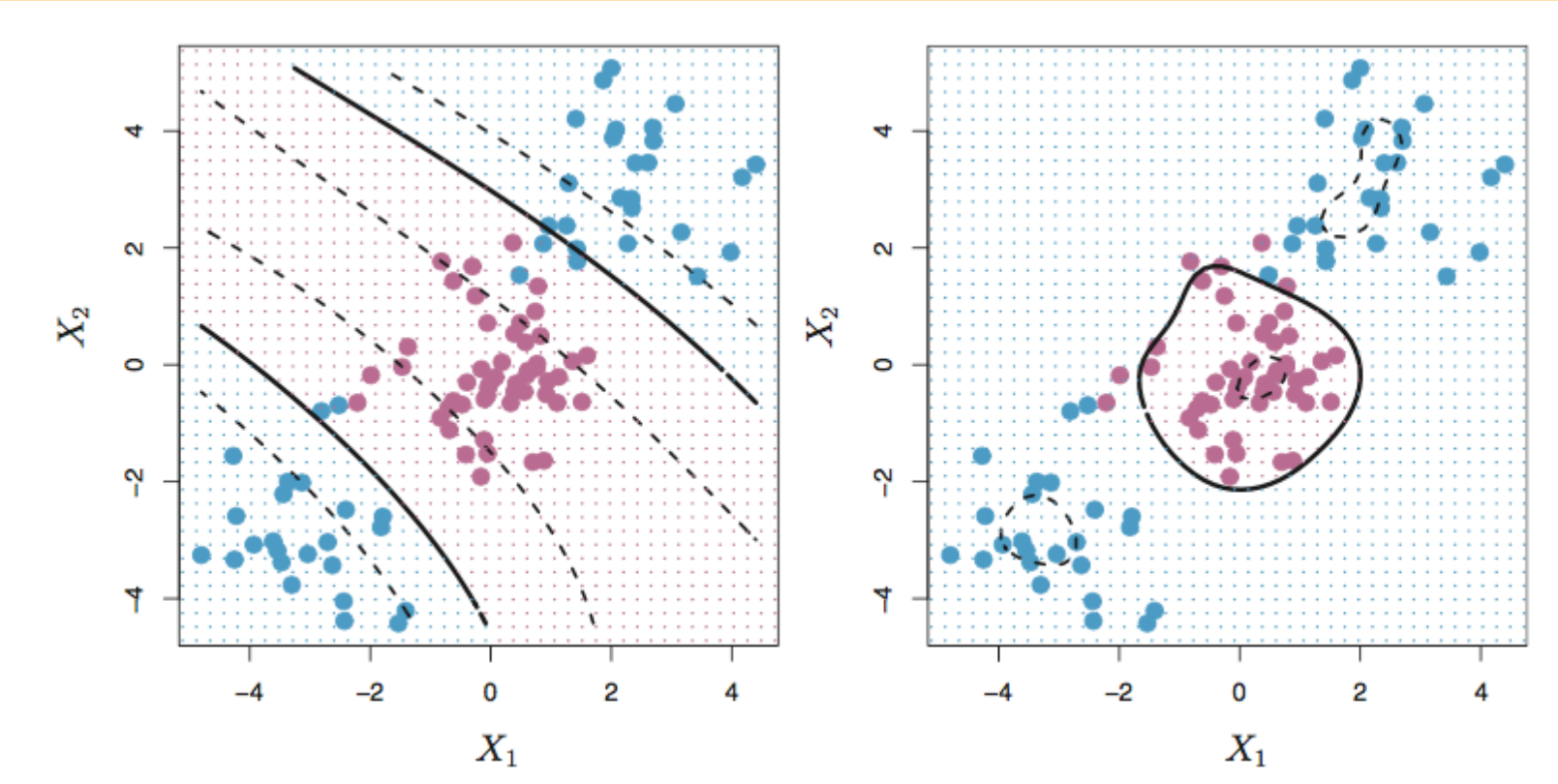


And linearity is limited

Example from ISLR

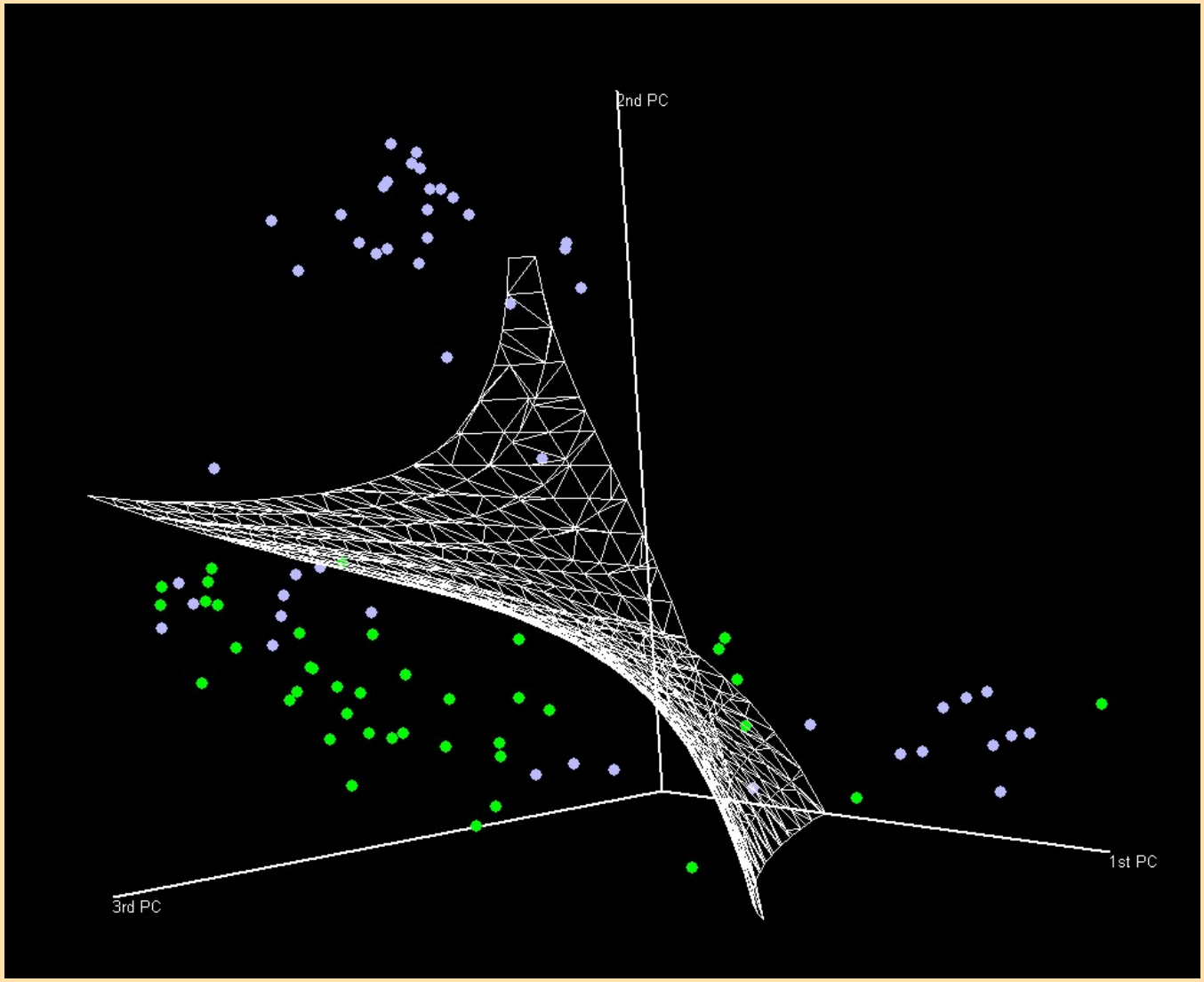


“Non-linear” decision boundary



The trick of SVM, is not to have non-linear models for the separation boundaries directly, but rather to warp the space that the data is in. That way, linear boundaries in the warped space become non-linear back in the original space.

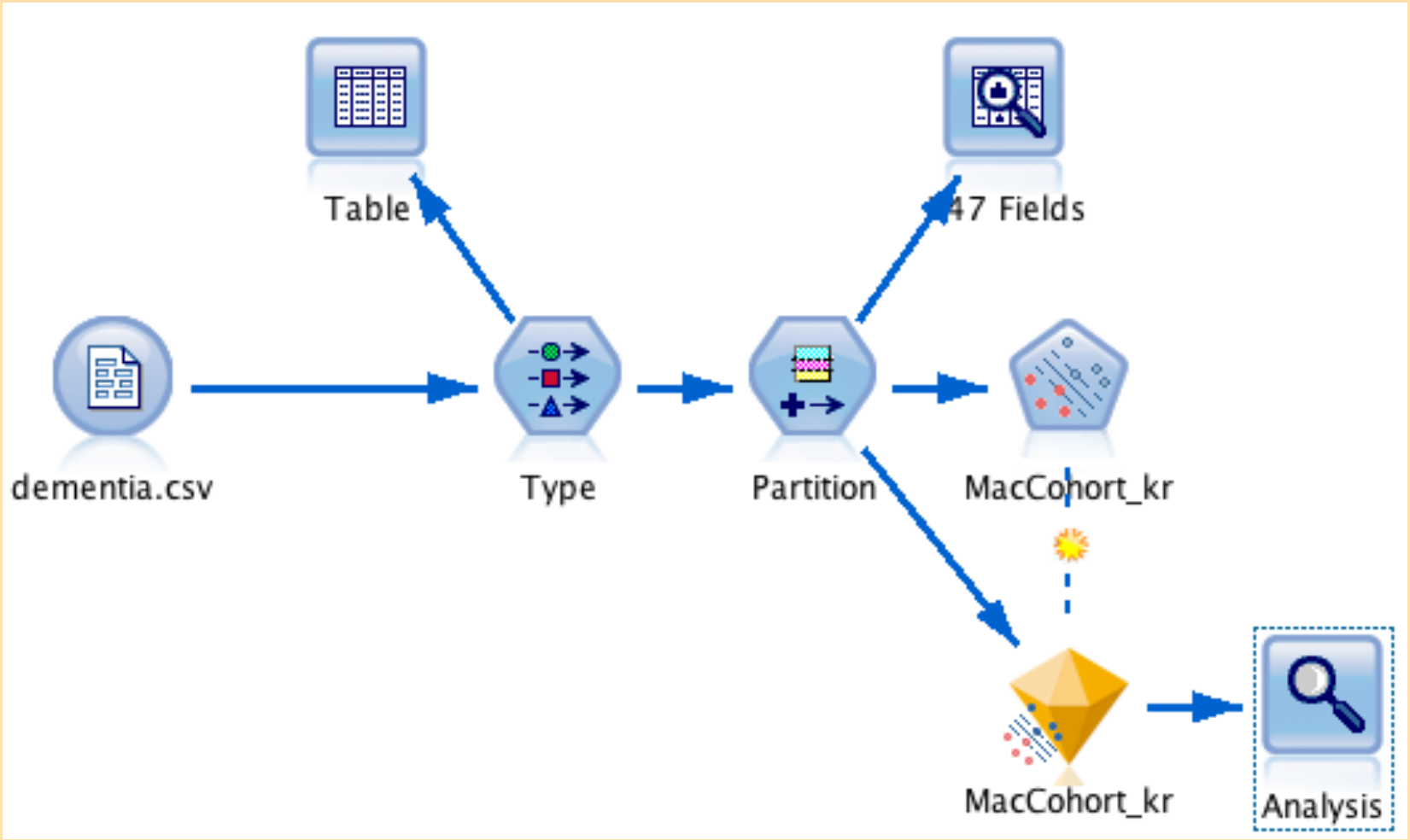
3 predictor – effective non-linear decision boundary



Support Vector Machine

- Considered one of the best “out of the box” classifiers for complex big data problems
- Has gained massive popularity
- Solved a lot of tough problems
- Great as a black box predictor, but not so good for interpretation

Modeler: Support Vector Machine



Support Vector Machine Node

MacCohort_kr

Fields | Model | Expert | Analyze | Annotations

Mode: Simple Expert

Append all probabilities (valid only for categorical targets)

Stopping criteria: 1.0E-3

Regularization parameter (C): 10

Regression precision (epsilon): 0.1

Kernel type: RBF

RBF gamma: 0.1

Bias: 0.0

Gamma: 1.0

Degree: 3

OK | Run | Cancel | Apply | Reset

Golden Nugget

Nothing much to see in here for multi-class problems

Analysis Node

Analysis of [MacCohort_kr]

File Edit

Analysis Annotations

Collapse All Expand All

Results for output field MacCohort_kr

Comparing $S-MacCohort_kr with MacCohort_kr

'Partition'	1_Training		2_Testing	
Correct	437	95.21%	156	77.61%
Wrong	22	4.79%	45	22.39%
Total	459		201	

Coincidence Matrix for $S-MacCohort_kr (rows show actuals)

'Partition' = 1_Training	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA	$null$
AD	100	2	0	0	0	0	3
CONTROL	0	210	0	0	0	0	4
PSP	1	1	30	0	0	0	3
bvFTD	0	1	0	42	0	0	2
nfPPAunspc	0	1	0	0	21	0	3
svPPA	0	0	0	0	0	34	1

'Partition' = 2_Testing	AD	CONTROL	PSP	bvFTD	nfPPAunspc	svPPA	$null$
AD	39	5	1	1	0	0	0
CONTROL	6	89	1	2	1	0	2
PSP	2	3	3	1	0	0	0
bvFTD	5	0	0	16	0	0	2
nfPPAunspc	3	2	3	3	1	0	1
svPPA	1	0	0	0	0	8	0

OK

Choosing a classifier

- **Logistic Regression**
  - Nice probabilistic interpretation
  - Not highly flexible
- **Kernel-bases classification**
  - No “model” required – don’t have to worry about outliers
  - Easy to understand rules
  - Not so good for interpretation
  - Not so good if you have very large number of predictors
- **Decision Trees**
  - Easy to interpret and explain
  - No “model” required -- don’t have to worry about outliers
  - Restricted to recursive partitioning of data to form classes
- **Support Vector Machines**
  - Often have high accuracy
  - Can deal with highly non-linear separation of features
  - Requires effort tuning and difficult to interpret
  - Black box – difficult to interpret
- **However, if your primary goal is good prediction, then you can run multiple classifiers to find the best, or even combine them to get an even better model overall (*ensemble* techniques of *boosting*, *bagging* and *stacking – hope to discuss Thursday*)**
- **If interested, take a look at last year’s Case Study lecture on the CLE. It performs an auto-classify node analysis of the dementia dataset using multiple classifiers**

Monday's lecture – clustering and data reduction

- Clustering
  - K-means clustering
  - Kohonen
- Data reduction
  - Principal components analysis (PCA)
- Guidance for choosing between Machine Learning methods
- Ensemble methods
- More classification techniques in brief (if time)

Note: HW #5 is up on CLE
– due 11.55pm on Thursday August 31