

# Hierarchical Predictive Analysis of Chemical Toxicity By Recursive Partitioning Using the Phase-I ToxCast Dataset

Jessie Q. Xia<sup>1</sup>  
S. Stanley Young<sup>1</sup>  
Russell S. Thomas<sup>2</sup>

1. National Institute of Statistical Sciences
2. The Hamner Institutes for Health Sciences

# OUTLINE

- Dataset
  - Multiblock
  - Augmented Atom Descriptors
- Methods and Results
  - Recursive partitioning
  - Hierarchical predictive analysis
- Discussion and Future Work

# ToxCast: MultiBlock Dataset

*In Vivo*  
Toxicity Data

*In Vitro*  
Biological Data

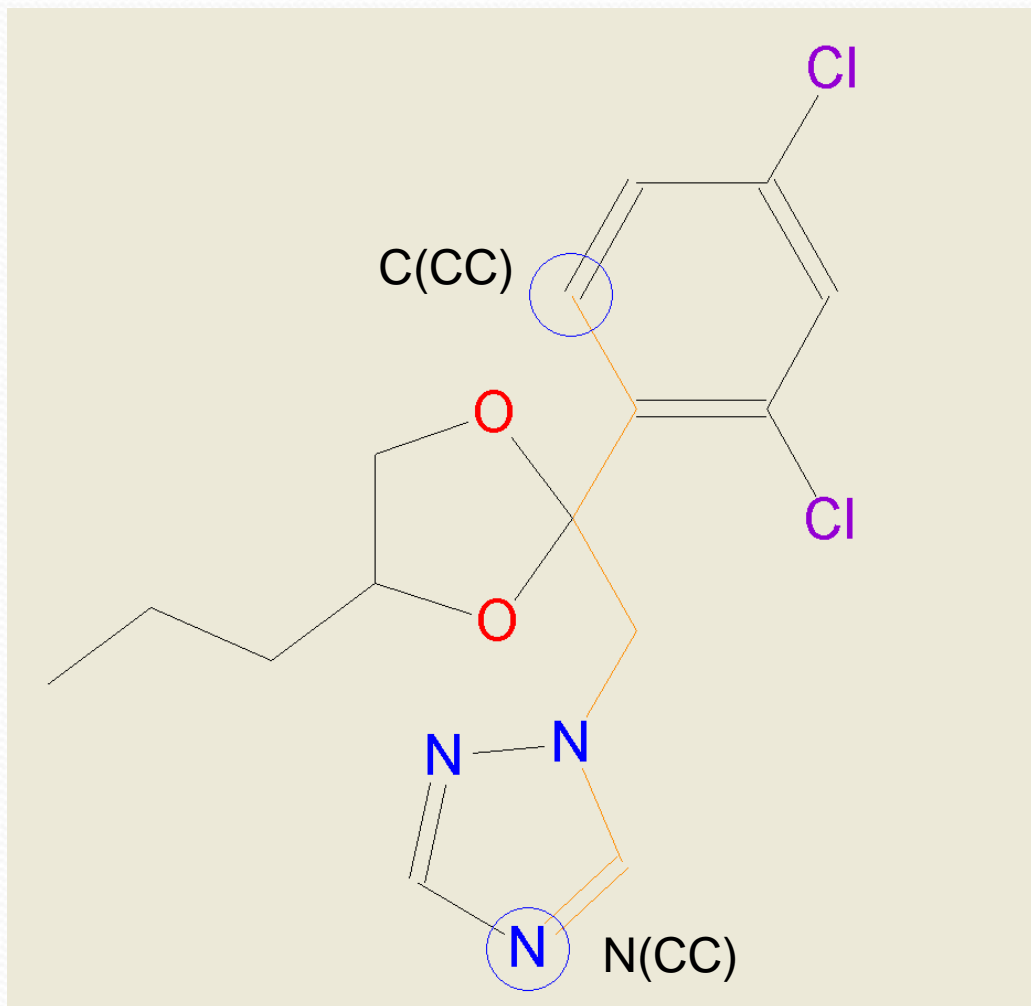
Physical  
Properties

Structure  
Features

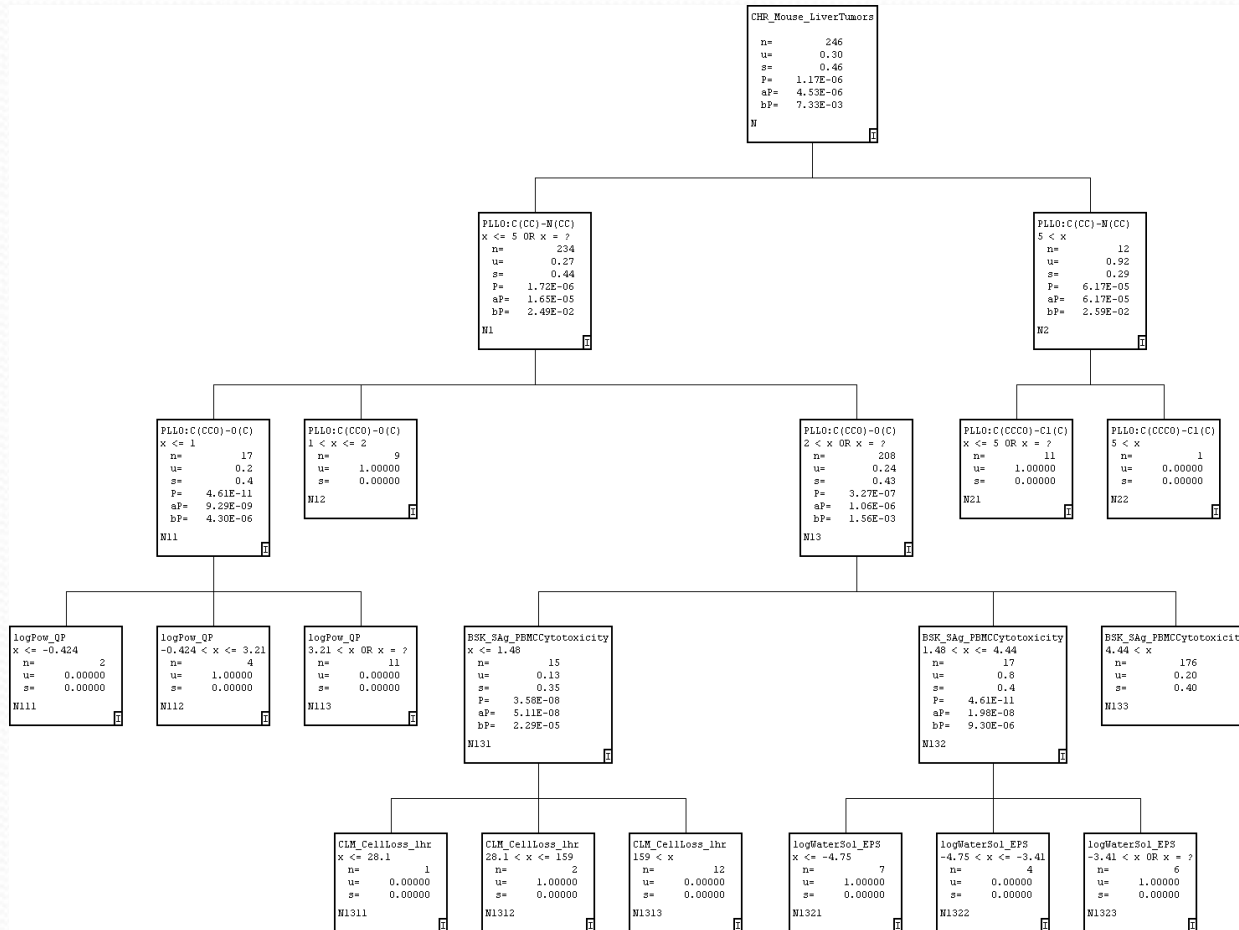
# Augmented Atom Descriptors

- Propiconazole

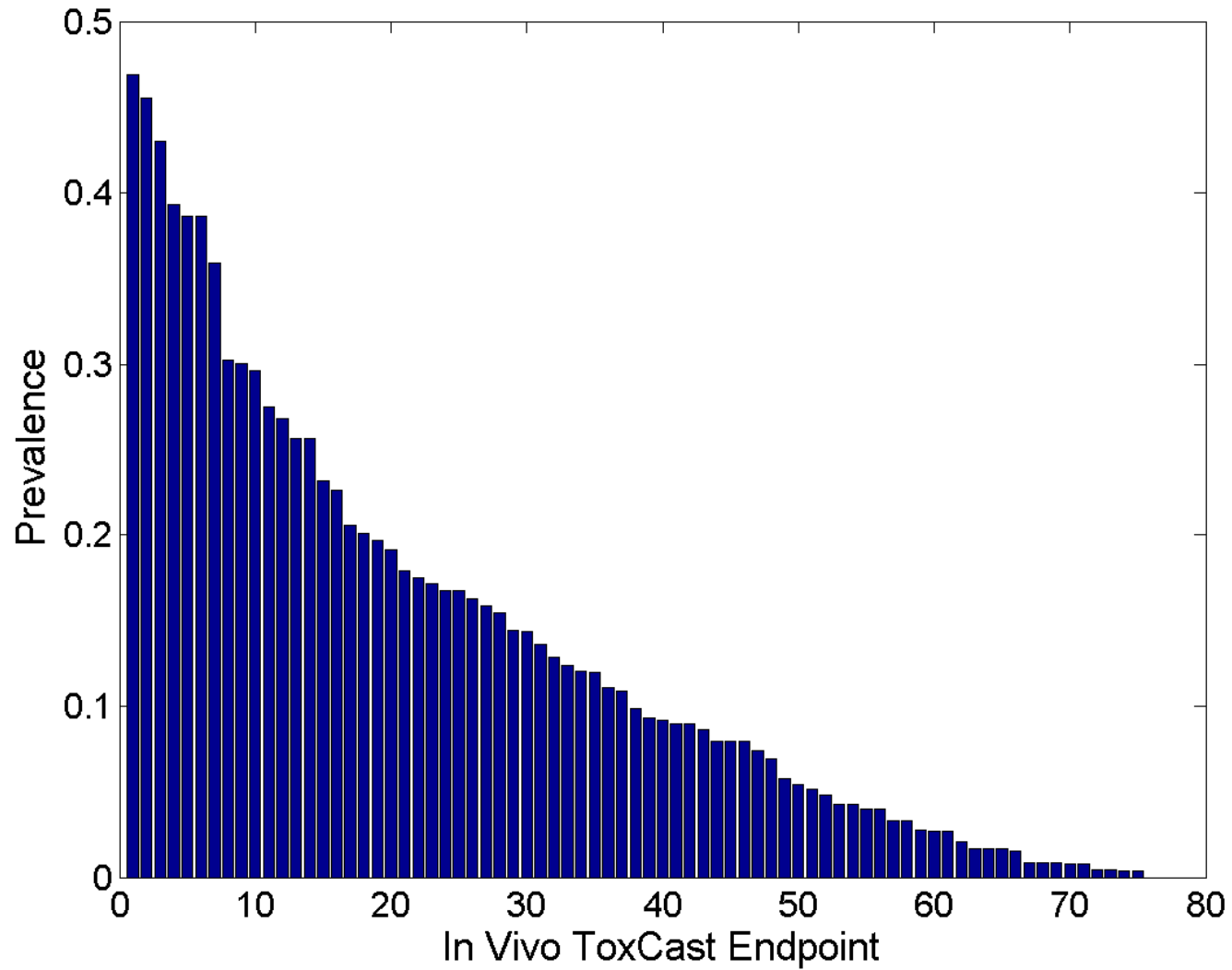
C(CC)-N(CC)



# Recursive Partitioning

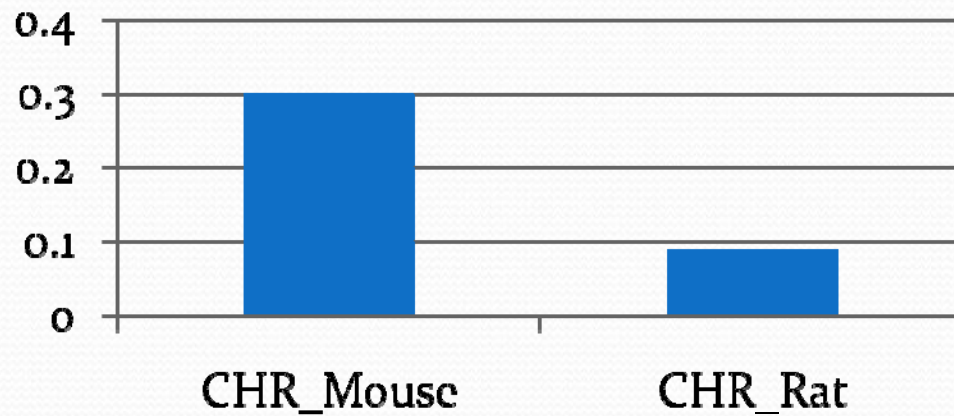


# Prevalence

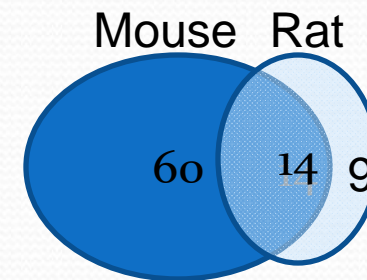


# Mouse Liver Tumors

## Prevalence



## Toxic Compounds

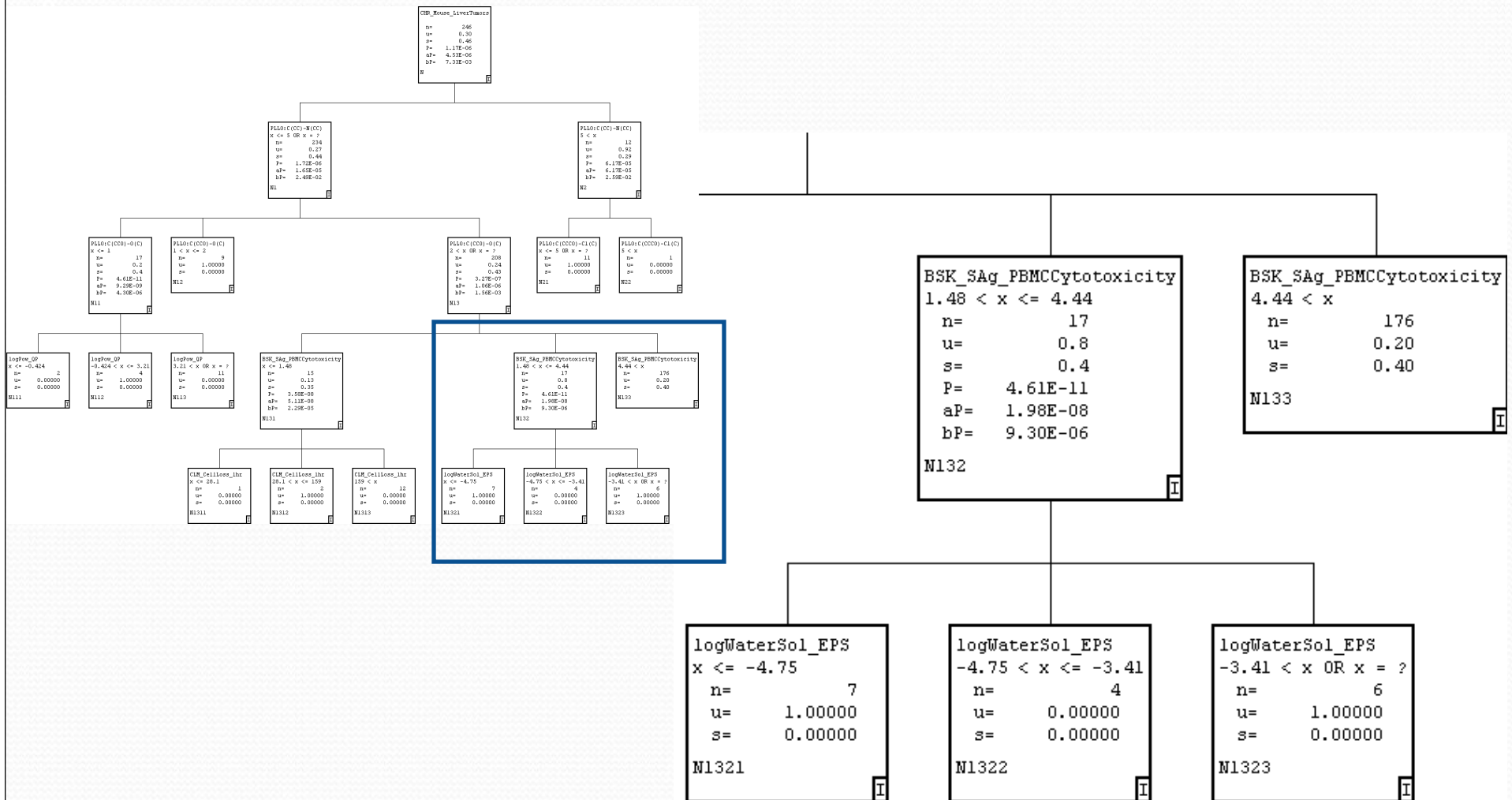


- Prediction across species
- Accuracy as a metric

# Performance of Predictive Models

- Sensitivity – Specificity Pairs
- Training vs. testing strategy
  - K-fold cross validation
- How representative is the dataset w.r.t. future data?
  - Predictive error

# Single RP Tree



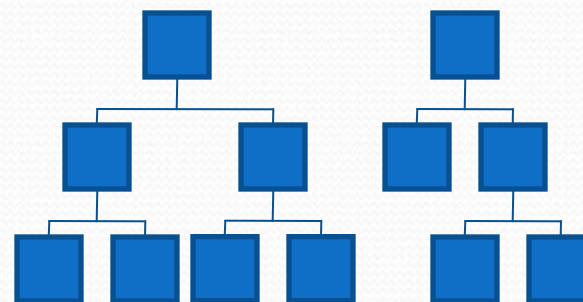
# Selected Predictors

Variables	Data Source
BSK_SAg_PBMCCytotoxicity	BioSeek
CLM_CellLoss_1hr	Cellumen
logPow_QP	QikProp
logWaterSol_EPS	EPISuite
PLLO:C(CC)-N(CC)	Atom Path Length
PLLO:C(CCCO)-Cl(C)	Atom Path Length
PLLO:C(CCO)-O(C)	Atom Path Length

# Multiple trees: Predictor Relations

- Metric:

$$\kappa = \frac{n_{ij} - \frac{n_i n_j}{n}}{\sqrt{\frac{n_i n_j}{n}}}$$



where

$n_{ij}$  – number of compounds under predictor i and j

$n_i$  – number of compounds under predictor i

$n_j$  – number of compounds under predictor j

- Criterion:

If  $\kappa > 0$  interaction between predictor i and j;

If  $\kappa = 0$  independence;

If  $\kappa < 0$  correlation;

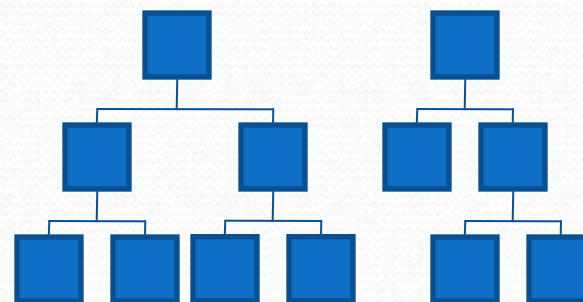
# Predictor Correlation &

	PLLO:C	BSK_SAg	volume_Q	PLLO:C(C	PLHI:C(C	logBB_Q	PLLO:C(C	CLM_Cel	PLLO:C(C	logPow_Q	K_Henry_
PLLO:C(CC)-N(CC)	0.96	0.79	0.051	0.36	0.36	0.0082	0.11	0.0076	0.1	0.0076	0.0077
BSK_SAg_PBMCCytotoxicity	0.3	0.8	0.051	0.36	0.35	0.01	0	0.0082	0.093	0	0.0061
volume_QP	0.1	0.8	0.051	0.022	0.029	0	0	0	0	0	0
PLLO:C(CCO)-C(COO)	-0.3	1.5	0.2	0.38	0	0.0035	0	0.0014	0	0	0.0021
PLHI:C(C)-C(CCCI)	0.1	1.3	1.1	-6.0	0.38	0.0062	0	0.0062	0	0	0.0028
logBB_QP	-0.3	0.4	-0.4	-0.1	0.6	0.01	0	0	0.00061	0	0
PLLO:C(CO)-O(C)	0.2	-4.7	-1.2	-3.3	-3.2	-0.5	0.11	0	0	0	0.0016
CLM_CellLoss_1hr	-0.1	0.3	-0.3	-0.5	0.9	-0.1	-0.5	0.0082	0.00061	0	0
PLLO:C(CCO)-O(C)	0.2	0.5	-1.1	-3.1	-3.1	-0.2	-1.7	-0.1	0.1	0.0076	0.0012
logPow_QP	0.1	-1.2	-0.3	-0.8	-0.8	-0.1	-0.5	-0.1	3.8	0.0076	0
K_Henry_EPS	0.1	-0.0	-0.3	-0.3	-0.0	-0.1	0.4	-0.1	0.2	-0.1	0.0077

# Multiple trees: Compound Similarity

- Metric:

$$\gamma = \frac{n_{ij}}{n}$$



where

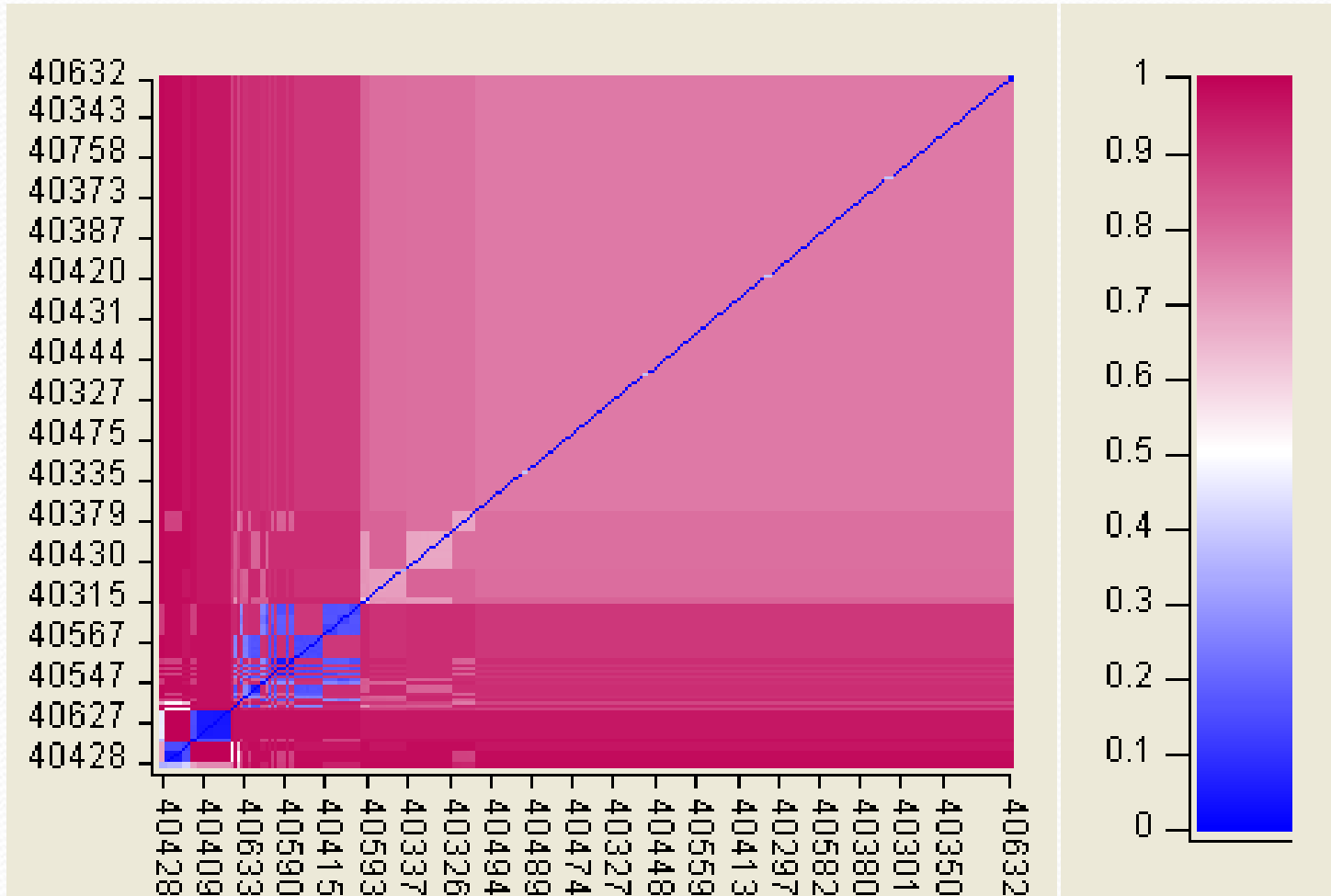
$n_{ij}$  – number of compounds in nodes  $i$  and  $j$

$n$  - total number of compounds in the root node

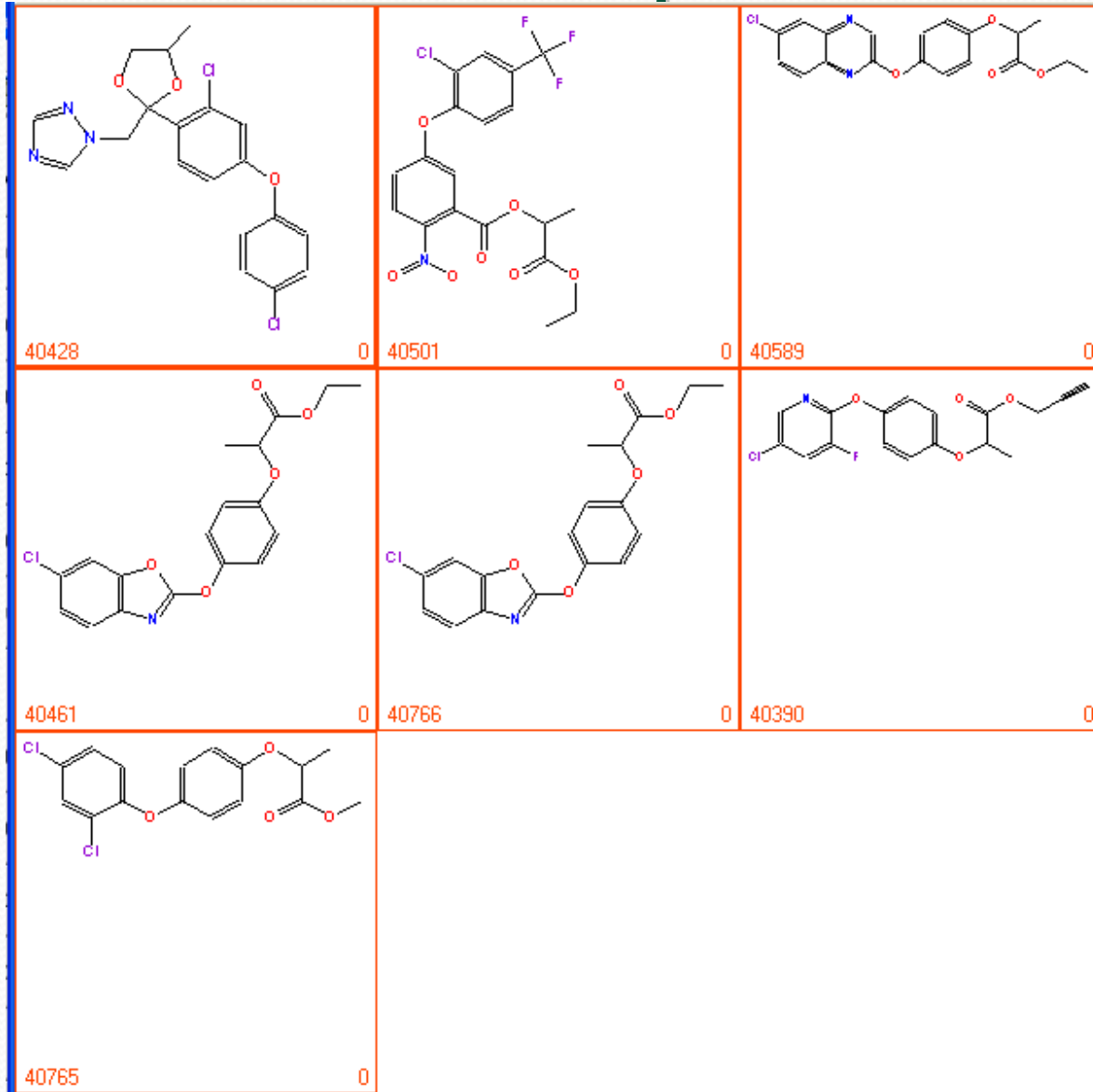
- Criterion:

If  $\gamma$  is smaller, the compounds are more similar.

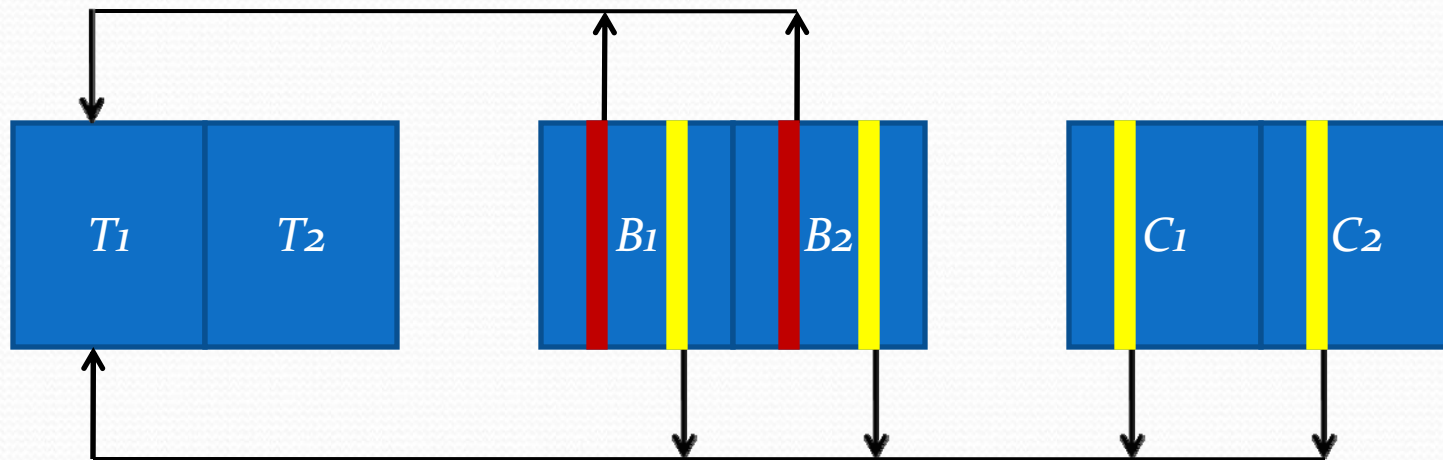
# Compound Similarity Matrix



# Similar Compounds



# Hierarchical Analysis



# Result

Metrics	Biological Data Only	Structural Data Only	Biological Data + Structural Data
Specificity	97.0%	100.0%	100.0%
Sensitivity	12.5%	6.3%	18.8%

# Discussion

- Recursive partitioning provides an intuitive explanation for the predictive analysis, and naturally separates multiple mechanisms.
- Recursive partitioning is a flexible tool with advanced features including predictor relationship exploration and compound similarity measure based on multiple trees.
- The performance of the recursive partitioning is improved by combining both the biological assay predictors and chemical descriptors.

# Future Work

- To tune the recursive partitioning algorithm such that it performs at the higher sensitivity at the expense of giving up some of specificity
- Not only predict 0-1 outcome, but also continuous toxicity data
- Evaluation of recursive partitioning w.r.t. additional machine learning algorithms available via
  - ChemModLab software on NCSU website
  - Advanced: Optimal Bit String Tree (ObsTree)

Thank you!