Time-series-based Ensemble Modeling for Bio-Medical Applications

Maciej Ogorzałek[1,2]

in collaboration with:

Christian Merkwirth, Grzegorz Surowka, Leszek Nowak, Katarzyna Grzesiak-Kopec [1]

Joerg Wichard [3]

[1] Department of Information Technologies, Jagiellonian University, Kraków

[2] Chair of Bio-signals and Systems, Hong Kong Polytechnic University (under DSS)

[3] FMP Berlin, Germany

Learning a Dependency from Data

Given: A sample of input-output-pairs $(\vec{x}^{\mu}, y^{\mu})$ with $\mu = 1, \ldots, N$
A functional dependence $y(\vec{x})$ (maybe corrupted by noise)

Aim: Choosing a model (function) $\hat{f}$ out of hypothesis space $\mathcal{H}$
close to true dependency f as possible

Classification $f : \mathbf{R}^D \mapsto \{0, 1, 2, ...\}$ discrete classes
Regression $f : \mathbf{R}^D \mapsto \mathbf{R}$ continuous output

Implementation usually via solution of an appropriate optimization problem:

- Matrix inversion in case of linear regression
- Minimization of a loss function on the training data
- Quadratic programming problem for SVMs

Validation and Model Selection

- Generalization error: How does the model perform on unseen data (samples) ?
- Exact generalization error is not accessible since we have only limited number of observations !
- Training on small data set tends to overfit, causing generalization error to be significantly higher than training error
- Consequence of mismatch between the capacity of the hypothesis space $\mathcal{H}$ (VC (Vapnik-Cervonenkis)-Dimension) and the number of training observations
- Validation: Estimating the generalization error using just the given data set
  - Needed for choosing optimal model structure or learning parameters (step sizes etc.)
- Model Selection: Selecting the model with lowest (estimated) generalization error
- But estimation of generalization error is very unreliable on small data sets

Improving Generalization for Single Models

- Remedies:
  - Manipulating training algorithm (e.g. early stopping)
  - Regularization by adding a penalty to the loss function
  - Using algorithms with built-in capacity control (e.g. SVM)
  - Rely on criteria like BIC (Bayesian Information Criteria), AIC (Akaike), GCV (Generalized Cross-Validation) or Cross Validation to select optimal model complextiy
  - Reformulate the loss function :
    - ϵ-insensitive loss
    - Huber loss
    - SVM loss for classification

Question

- Are there any other methods to improve generalization error ?

Question

- Are there any other methods to improve generalization error ?
- Yes, by combining several individual models!

Ensemble Methods

Ensemble: Averaging the output of several separately trained models

- Simple average
  $\bar{f}(\vec{x}) = \frac{1}{K} \sum_{k=1}^{K} f_k(\vec{x})$
- Weighted average
  $\bar{f}(\vec{x}) = \sum_k w_k f_k(\vec{x})$ with $\sum_k w_k = 1$

Ensemble Methods

Ensemble: Averaging the output of several separately trained models

- Simple average
  $\bar{f}(\vec{x}) = \frac{1}{K} \sum_{k=1}^{K} f_k(\vec{x})$
- Weighted average
  $\bar{f}(\vec{x}) = \sum_k w_k f_k(\vec{x})$ with $\sum_k w_k = 1$

Ensemble Methods

Ensemble: Averaging the output of several separately trained models

- Simple average
  $\bar{f}(\vec{x}) = \frac{1}{K} \sum_{k=1}^{K} f_k(\vec{x})$
- Weighted average
  $\bar{f}(\vec{x}) = \sum_k w_k f_k(\vec{x})$ with $\sum_k w_k = 1$

Ensemble Methods

Ensemble: Averaging the output of several separately trained models

- Simple average
  $\bar{f}(\vec{x}) = \frac{1}{K}\sum_{k=1}^{K} f_k(\vec{x})$
- Weighted average
  $\bar{f}(\vec{x}) = \sum_k w_k f_k(\vec{x})$ with $\sum_k w_k = 1$

Interpretation:

- The ensemble generalization error is always smaller than the expected error of the individual models
- An ensemble should consist of well trained but diverse models
- An ensemble often outperforms the best constituting model

Error decomposition:

$$
\begin{aligned}
e(\vec{x}) &= (y(\vec{x}) - \bar{f}(\vec{x}))^2 \\
\bar{\epsilon}(\vec{x}) &= \frac{1}{K}\sum_{k=1}^{K}(y(\vec{x}) - f_k(\vec{x}))^2 \\
\bar{a}(\vec{x}) &= \frac{1}{K}\sum_{k=1}^{K}(f_k(\vec{x}) - \bar{f}(\vec{x}))^2
\end{aligned}
$$

$$
e(\vec{x}) = \bar{\epsilon}(\vec{x}) - \bar{a}(\vec{x})
$$

Integrating over input space:

$$
\mathbf{E} = \bar{\mathbf{E}} - \bar{\mathbf{A}}
$$

Decorrelating Models

$$\mathbf{E} = \bar{\mathbf{E}} - \bar{\mathbf{A}}$$

How can we obtain models that have low generalization error (small $\bar{E}$), but are mutually uncorrelated (large $\bar{A}$)?

- Varying model structure (e.g. topology)
- Exploiting the disadvantage of getting stuck in local minima:
  - Varying initial conditions
  - Varying parameters of the training procedure
  - Using ϵ-insensitive loss function
- Train a large population of models
- Applying resampling or sequencing techniques:

Decorrelating Models

$$\mathbf{E} = \bar{\mathbf{E}} - \bar{\mathbf{A}}$$

How can we obtain models that have low generalization error (small $\bar{E}$), but are mutually uncorrelated (large $\bar{A}$)?

- Varying model structure (e.g. topology)
- Exploiting the disadvantage of getting stuck in local minima:
  - Varying initial conditions
  - Varying parameters of the training procedure
  - Using ϵ-insensitive loss function
- Train a large population of models
- Applying resampling or sequencing techniques:

- Resampling: Generating new data sets by omitting or duplicating samples of the original data set. These techniques can be used to estimate generalization errors and for model construction

  Bootstraping Generate bootstrap replicates by randomly drawing samples from training set

  Cross-Validation Divide data set repeatedly in training and test part

  Bumping Construct models on bootstrap replicates and choose best model on full data set

  Bagging Bootstrap aggregation, create several models on bootstrap replicates and average these

  Boosting Create sequence of models where training of next model depends on output of previous model

Crosstraining – Constructing Ensembles

- Finesse: Efficiently reuse samples by combining training, validation and selection of models
- Additional benefit of reduced correlation between models
- Repeatedly partition data set randomly into two sample classes
  - Training set, used for training and stopping criteria
  - Test set, used only for accessing generalization error after model has been trained

Crosstraining – Constructing Ensembles

- Finesse: Efficiently reuse samples by combining training, validation and selection of models
- Additional benefit of reduced correlation between models
- Repeatedly partition data set randomly into two sample classes
  - Training set, used for training and stopping criteria
  - Test set, used only for accessing generalization error after model has been trained
- Train population of (heterogenous) models, select best ones according to error on test set
- Repartition data set, taking care that test sets are mutually disjunct
- Combine best models of all partitionings to ensemble
- Optionally weight models according to the estimated generalization error on the total data set

Pros and Cons of Ensembles

Ensemble Methods

- Advantages
  - Straightforward extension of existing modeling algorithms
  - Almost fool-proof minimization of generalization error
  - Makes no assumptions on the structure of the underlying models
  - Simplifies the problem of model selection
- Disadvantages
  - Increased computational effort
  - Interpretation of ensemble is even harder than drawing conclusions from a single model

Pros and Cons of Ensembles

Ensemble Methods

- Advantages
  - Straightforward extension of existing modeling algorithms
  - Almost fool-proof minimization of generalization error
  - Makes no assumptions on the structure of the underlying models
  - Simplifies the problem of model selection
- Disadvantages
  - Increased computational effort
  - Interpretation of ensemble is even harder than drawing conclusions from a single model

Combining Heterogenous Models

- Advantages
  - Often one model type performs superior on the given data set
  - Probability of using an unsuited model type decreases
  - Inherent decorrelation even without manipulating data set or training parameters
- Disadvantages
  - Accessing the generalization performance of heterogenous models is even more difficult than for models of same type

The ENTOOL Toolbox for Statistical Learning

- The ENTOOL toolbox for statistical learning is designed to make state-of-the-art machine learning algorithms available under a common interface
- Allows construction of single models or ensembles of (heterogenous) models
- Supports decorrelation of models by offering resampling techniques
- Though primarily designed for regression, it is possible to construct ensembles of classifiers with ENTOOL

The ENTOOL Toolbox for Statistical Learning

- The ENTOOL toolbox for statistical learning is designed to make state-of-the-art machine learning algorithms available under a common interface
- Allows construction of single models or ensembles of (heterogenous) models
- Supports decorrelation of models by offering resampling techniques
- Though primarily designed for regression, it is possible to construct ensembles of classifiers with ENTOOL

- Requirements:
  - Matlab (TM)
- Operating systems:
  - Windows
  - Linux
  - Solaris (limited)

ENTOOL Software Architecture

- Each model type is implemented as separate class
- All model classes share common interface
- Exchange model types by exchanging constructor call
- Automatic generation of ensembles of models
- Models are divided into two brands:
  1. Primary models like linear models, neural networks, SVMs etc.
  2. Secondary models that rely on primary models to calculate output. All ensemble models are secondary models.

ENTOOL Software Architecture

- Each model type is implemented as separate class
- All model classes share common interface
- Exchange model types by exchanging constructor call
- Automatic generation of ensembles of models
- Models are divided into two brands:
  1. Primary models like linear models, neural networks, SVMs etc.
  2. Secondary models that rely on primary models to calculate output. All ensemble models are secondary models.

- Lifecycle of a model can be divided into three phases:
  1. During construction, topology of the model is specified. The model can't be used yet.
  2. Model has now to be trained on some training data set $(\vec{x}_i, y_i)$
  3. After training, the model can be evaluated on new/unseen inputs $(\vec{x}_n)$
- Constructors should assign random default topologies in order to create uncorrelated models
- It is possible to construct ensembles of ensembles

Syntax

- Constructor syntax:
  `model = perceptron;` creates a MLP model with default topology
  `model = perceptron(12);` MLP model with 12 hidden layer neurons
  `model = ridge;` creates a linear model by ridge regression

Syntax

- Constructor syntax:
  `model = perceptron;` creates a MLP model with default topology
  `model = perceptron(12);` MLP model with 12 hidden layer neurons
  `model = ridge;` creates a linear model by ridge regression
- Training syntax:
  `model = train(model, x, y, [], [], 0.05);`
  trains model with ϵ-insensitive loss of 0.05 on data set $(\vec{x}_i, y_i)$

Syntax

- Constructor syntax:
  `model = perceptron;` creates a MLP model with default topology
  `model = perceptron(12);` MLP model with 12 hidden layer neurons
  `model = ridge;` creates a linear model by ridge regression
- Training syntax:
  `model = train(model, x, y, [], [], 0.05);`
  trains model with ϵ-insensitive loss of 0.05 on data set $(\vec{x}_i, y_i)$
- Evaluation syntax:
  `y_new = calc(model, x_new)` evaluates the model on new inputs

Syntax

- Constructor syntax:
  `model = perceptron;` creates a MLP model with default topology
  `model = perceptron(12);` MLP model with 12 hidden layer neurons
  `model = ridge;` creates a linear model by ridge regression
- Training syntax:
  `model = train(model, x, y, [], [], 0.05);`
  trains model with ϵ-insensitive loss of 0.05 on data set $(\vec{x}_i, y_i)$
- Evaluation syntax:
  `y_new = calc(model, x_new)` evaluates the model on new inputs
- How to build an ensemble of models:
  `ens = crosstrainensemble;` will create an empty ensemble object
  `ens = train(ens, x, y, [], [], 0.05);` calls training routines for several primary models and joins them into ensemble object

Adjusting class specific training parameters

- 5th argument when calling `train` specifies training parameters

Adjusting class specific training parameters

- 5th argument when calling `train` specifies training parameters
- Except topology, often training parameters have to be specified:

```
tp = get(perceptron, 'trainparams')
error_loss_margin:  0.0100
decay:  0.0010
rounds:  500
mrate_init:  0.0100
max_weight:  10
mrate_grow:  1.2000
mrate_shrink:  0.5000
```

Adjusting class specific training parameters

- 5th argument when calling `train` specifies training parameters
- Except topology, often training parameters have to be specified:

```
tp = get(perceptron, 'trainparams')
error_loss_margin:  0.0100
decay:  0.0010
rounds:  500
mrate_init:  0.0100
max_weight:  10
mrate_grow:  1.2000
mrate_shrink:  0.5000
```

- Assign new value: `tp.decay = 0.05`

Adjusting class specific training parameters

- 5th argument when calling `train` specifies training parameters
- Except topology, often training parameters have to be specified:

```
tp = get(perceptron, 'trainparams')
error_loss_margin:  0.0100
decay:  0.0010
rounds:  500
mrate_init:  0.0100
max_weight:  10
mrate_grow:  1.2000
mrate_shrink:  0.5000
```

- Assign new value: `tp.decay = 0.05`
- And give training parameters while training:

```
model = train(perceptron, x, y, [], tp, 0.05);
```

Specifying which Model Types to Ensemble

- Ensemble constructor will train several models on dataset:

```
tp = get(crosstrainensemble, 'trainparams')
nr_cv_partitions:  8
frac_test:  0.2000
minimum_testsamples:  5
remove_worst:  0.3300
use_models:  0.8000
weight_models:  0
modelclasses:  6x3 cell
scaledata:  1
```

Specifying which Model Types to Ensemble

- Ensemble constructor will train several models on dataset:

```
tp = get(crosstrainensemble, 'trainparams')
nr_cv_partitions:  8
frac_test:  0.2000
minimum_testsamples:  5
remove_worst:  0.3300
use_models:  0.8000
weight_models:  0
modelclasses:  6x3 cell
scaledata:  1
```

- Assign new value:

```
tp.modelclasses = {'perceptron', [], {}; ...
{'lssvm', [], {'function', 'RBF_kernel', 100, 2}}
```

Specifying which Model Types to Ensemble

- Ensemble constructor will train several models on dataset:

```
tp = get(crosstrainensemble, 'trainparams')
nr_cv_partitions:  8
frac_test:  0.2000
minimum_testsamples:  5
remove_worst:  0.3300
use_models:  0.8000
weight_models:  0
modelclasses:  6x3 cell
scaledata:  1
```

- Assign new value:

```
tp.modelclasses = {'perceptron', [], {}; ...
{'lssvm', [], {'function', 'RBF_kernel', 100, 2}}
```

- And give training parameters while training:

```
ens = train(crosstrainensemble, x, y, [], tp, 0.05);
```

Primary Models Types

ares Adaption of Friedman's MARS algorithm

ridge Linear model based on ridge regression with implicit LOO cross-validation for selecting optimal ridge penalty

perceptron Multilayer perceptron with iRPROP+ training

perceptron2 Magnus Nørgaard's single layer perceptron, trained with Levenberg-Marquart

prbfn Shimon Cohen's projection based radial basis function network

rbf Mark Orr's radial basis function code

vicinal k-nearest-neighbor regression with adaptive metric

mpmr Thomas Strohmann's Mimimax Probability Machine Regression

lssvm Johan Suykens' least-square SVM toolbox

tree Adaption of Matlab's build-in regression/classification trees

osusvm SVM code based on Chih-Jen Lin's libSVM

vicinalclass k-nearest-neighbor classification

Ensemble Classes

ensemble Virtual parent class for all ensemble classes

crosstrainensemble Ensemble class that trains models according to crosstraining scheme. Creates ensembles of decorrelated models.

cvensemble Ensemble class that trains models according to crossvalidation/out-of-training scheme. Can be used to access OOT error.

extendingsetensemble Boosting variant for regression.

subspaceensemble Creates an ensemble of models where each single model is trained on a random subspace of the input data set.

optimalsvm Wrapper that trains RBF osusvm/lssvm with optimal parameter settings (C and γ)

featureselector Does feature selection and trains model on selected subset

ENTOOL webpage

http://zti.if.uj.edu.pl/ merkwirth/entool.htm

Application Examples

- Applications using ENTOOL
  - Nonlinear Regression of Skin Permeability
  - Sequence Analysis
- Molecular Graph Networks
  - Classification on NCI Data Set
  - Regression on KDD Challenge Data
  - Skin Cancer diagnosis

Receiver Operating Characteristics

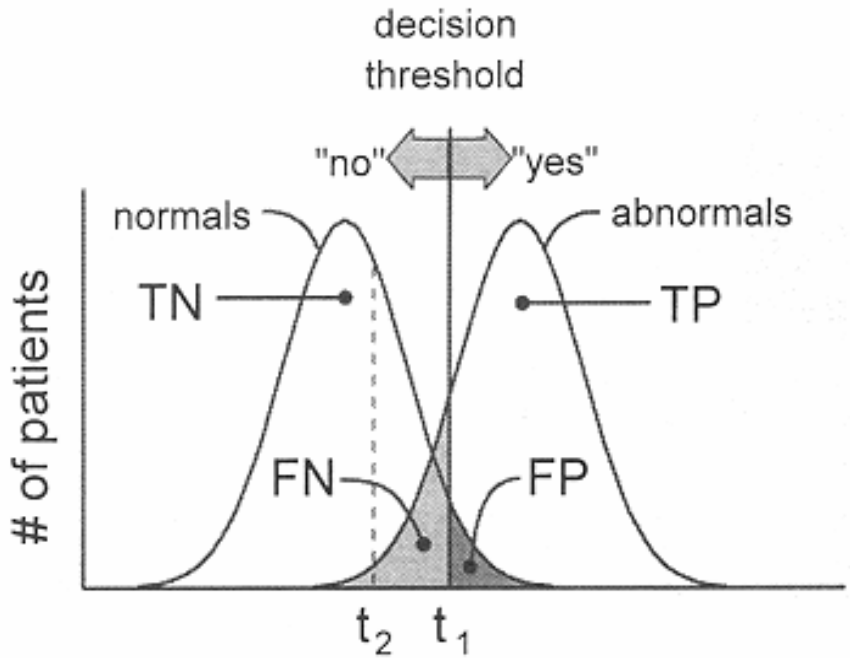
- Most basic task of the diagnostician is to separate abnormal subjects from normal subjects
- In many cases there is significant overlap in terms of the appearance of the image
  - Some abnormal patients are normal-looking
  - Some normal patients are abnormal-looking

2 x 2 decision matrix

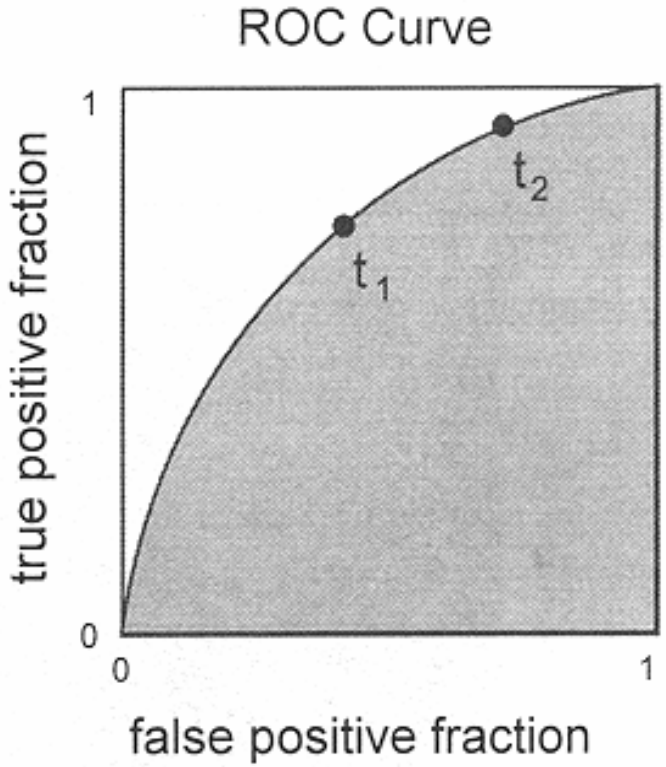
	Actually Abnormal	**Actually Normal**
Diagnosed as Abnormal	True Positive (TP)	False Positive (FP)
Diagnosed as Normal	False Negative (FN)	True Negative (TN)

ROC curves (cont.)

- For a single threshold value and the population being studied, a single value for TP, TN, FP, and FN can be computed
- The sum TP + TN + FP + FN will be equal to the total number of normals and abnormals in the study population
- “True” diagnosis must be determined independently, based on biopsy confirmation, long-term patient follow-up, etc.


decision
threshold
"no"
"yes"
normals
abnormals
TN
TP
FN
FP
# of patients
t_2 t_1
decision criterion

A


ROC Curve
1
0
t_2
t_1
true positive fraction
0
1
false positive fraction

B

ROC curves (cont.)

- True-positive fraction (TPF) = TP/(TP + FN)
- False-positive fraction (FPF) = FP/(FP + TN)
- A ROC curve is a plot of the true-positive fraction versus the false-positive fraction. A single threshold value will produce a single point on the ROC curve
- In practice, 5 points are realized based on the confidence level of the observer (*definitely there, maybe there, uncertain, maybe not there,* and *definitely not there*)

Sensitivity and specificity

- *Sensitivity* is the fraction of abnormal cases that a decision maker actually calls abnormal:

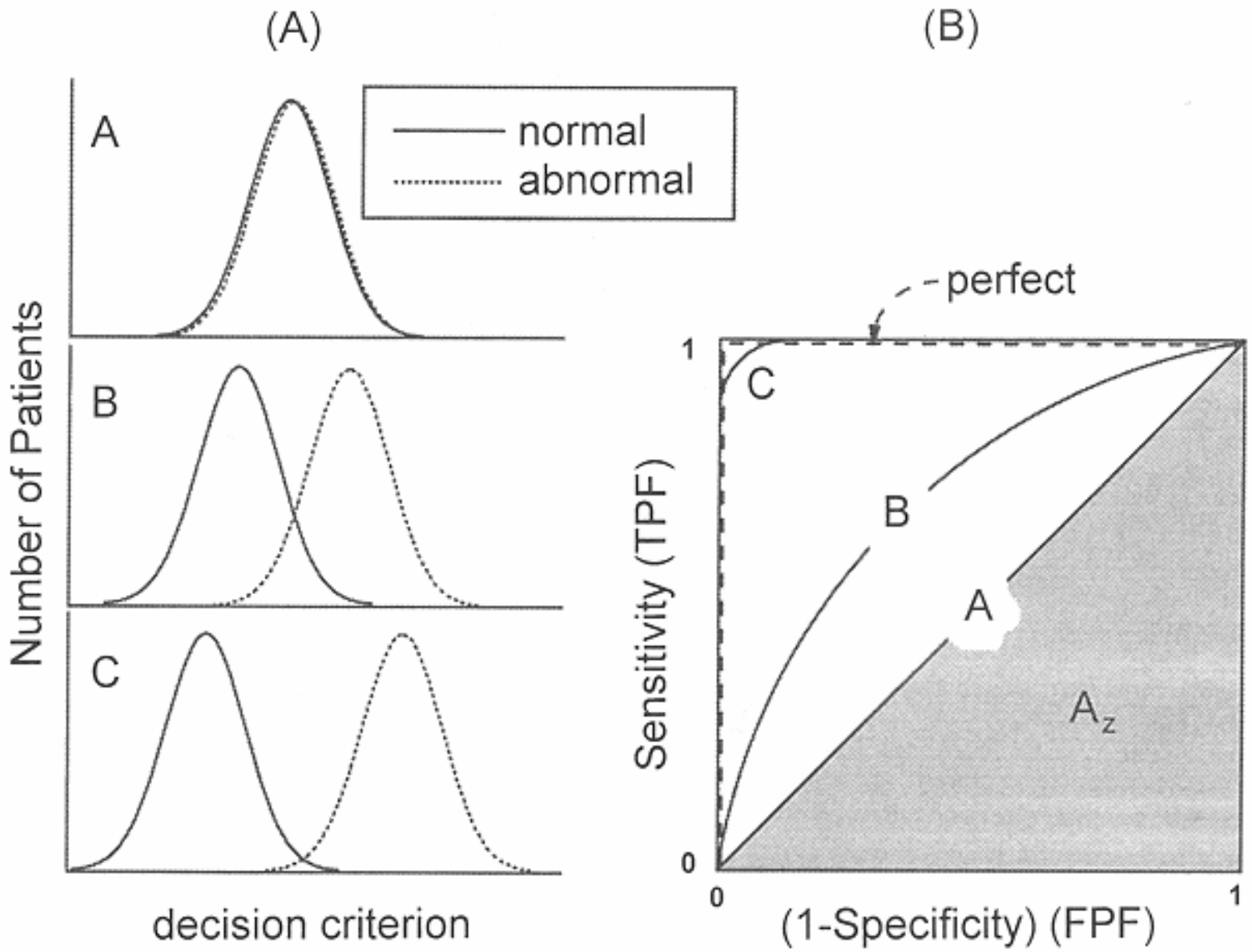
$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP+FN}}$$

- *Specificity* is the fraction of normal cases that a decision maker actually calls normal:

$$\text{Specificity} = \frac{\text{TN}}{\text{TN+FP}}$$

Interpretation

- An ROC curve is essentially a way of analyzing the SNR associated with a certain diagnostic task
- In addition to the inherent SNR of the imaging modality under investigation, different human observers have *internal noise*, which affects individual performance
- Different radiologists may have different ROC curves


(A)
(B)
A
B
C
normal
abnormal
Number of Patients
decision criterion
perfect
1
0
C
B
A
A_z
Sensitivity (TPF)
0
1
(1-Specificity) (FPF)

Interpretation (cont.)

- Set A has almost complete overlap between abnormal and normal cases
  - The SNR is near zero; ROC curve A represents *pure guessing* in terms of the diagnosis
- As separation between normal and abnormal cases increases (sets B & C), the corresponding ROC curves approach the upper left corner
- Area under the ROC curve is a measure of *detectability*
  - For worst performance, $A_Z = 0.5$
  - For best performance, $A_Z = 1.0$

Sensitivity Analysis for Regression

- Motivation: Determine variable importance with respect to prediction accuracy
- Might help uncovering causal relationships of underlying process
- Problem: Ensemble of heterogenous (nonlinear) models is even more difficult to analyze than single models
- Idea: Combine surrogate data method with OOT calculation

Sensitivity Analysis for Regression

- Motivation: Determine variable importance with respect to prediction accuracy
- Might help uncovering causal relationships of underlying process
- Problem: Ensemble of heterogenous (nonlinear) models is even more difficult to analyze than single models
- Idea: Combine surrogate data method with OOT calculation

- To determine importance of n-th variable:
  - Create surrogate/replicate of the original input data set where values of n-th variable are permuted randomly to destroy information content
  - Calculate OOT output for surrogate data set
  - Compare errors of OOT output of surrogate and original data set
  - If OOT error increases significantly, the n-th variable is important!
  - Average importance over several surrogate data sets for same variable to smooth out noise

Sensitivity Analysis for Regression

- Motivation: Determine variable importance with respect to prediction accuracy
- Might help uncovering causal relationships of underlying process
- Problem: Ensemble of heterogenous (nonlinear) models is even more difficult to analyze than single models
- Idea: Combine surrogate data method with OOT calculation
- Retraining unnecessary, would mask importance of correlated inputs
- Uncovers linear and nonlinear relationships

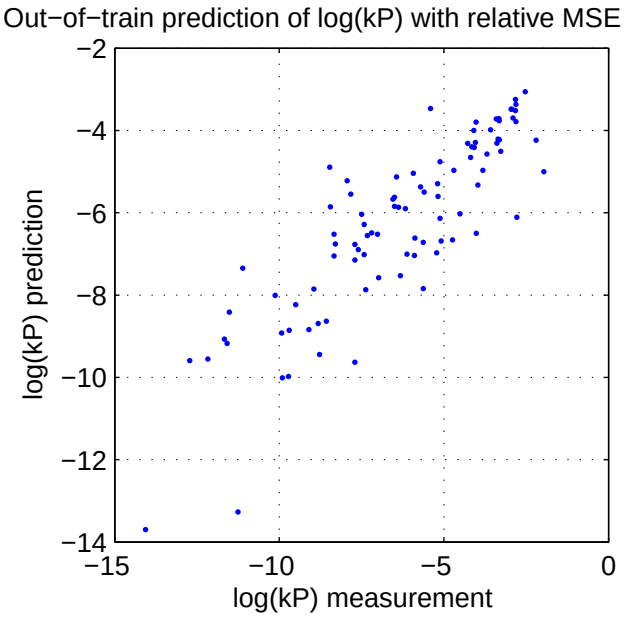
- To determine importance of n-th variable:
  - Create surrogate/replicate of the original input data set where values of n-th variable are permuted randomly to destroy information content
  - Calculate OOT output for surrogate data set
  - Compare errors of OOT output of surrogate and original data set
  - If OOT error increases significantly, the n-th variable is important!
  - Average importance over several surrogate data sets for same variable to smooth out noise

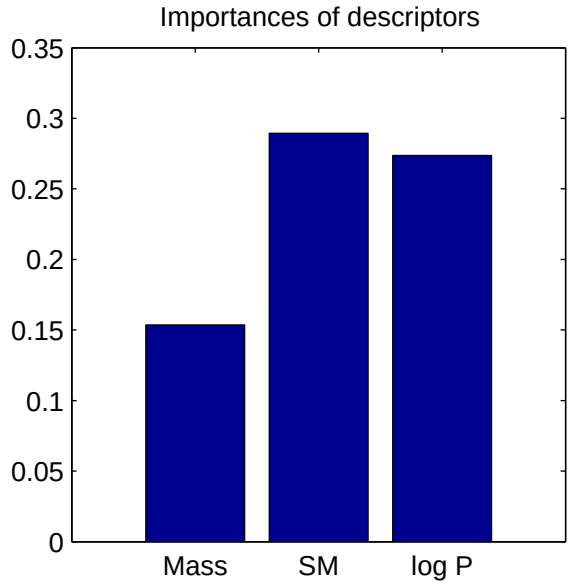
Nonlinear Regression of Skin Permeability

- 93 compounds described by 131 descriptors
- Ensemble of linear ridge models and k-nearest neighbor models
- Identified 8 descriptors by sensitivity analysis: ’Mass’ ’Log P (oct/wat)’ ’Cosmo’ ’weinerPol’ ’logP(o/w)’ ’SM 5.0R’ ’TPSA’ ’vol’
- Exhaustive check of all combinations of these descriptors leads to two final models:
  - ’Mass’ ’logP(o/w)’ ’Cosmo’ with OOT error on training data set of 0.30 RMSE and on validation set of 0.31 RMSE
  - ’Mass’ ’SM 5.0R’ ’Log P (oct/wat)’ with RMSE 0.28/0.28

Nonlinear Regression of Skin Permeability

- 93 compounds described by 131 descriptors
- Ensemble of linear ridge models and k-nearest neighbor models
- Identified 8 descriptors by sensitivity analysis: ’Mass’ ’Log P (oct/wat)’ ’Cosmo’ ’weinerPol’ ’logP(o/w)’ ’SM 5.0R’ ’TPSA’ ’vol’
- Exhaustive check of all combinations of these descriptors leads to two final models:
  - ’Mass’ ’logP(o/w)’ ’Cosmo’ with OOT error on training data set of 0.30 RMSE and on validation set of 0.31 RMSE
  - ’Mass’ ’SM 5.0R’ ’Log P (oct/wat)’ with RMSE 0.28/0.28





Sensitivity Analysis for Sequence Analysis

- Motivation: Determine importance of amino acid positions with respect to genotype-phenotype prediction accuracy
- Same idea as the sensitivity analysis for regression, but:
  - decrease in AUC (area under curve in ROC plot) instead of increase of MSE
  - random permutation of amino acids for each position

Sensitivity Analysis for Sequence Analysis

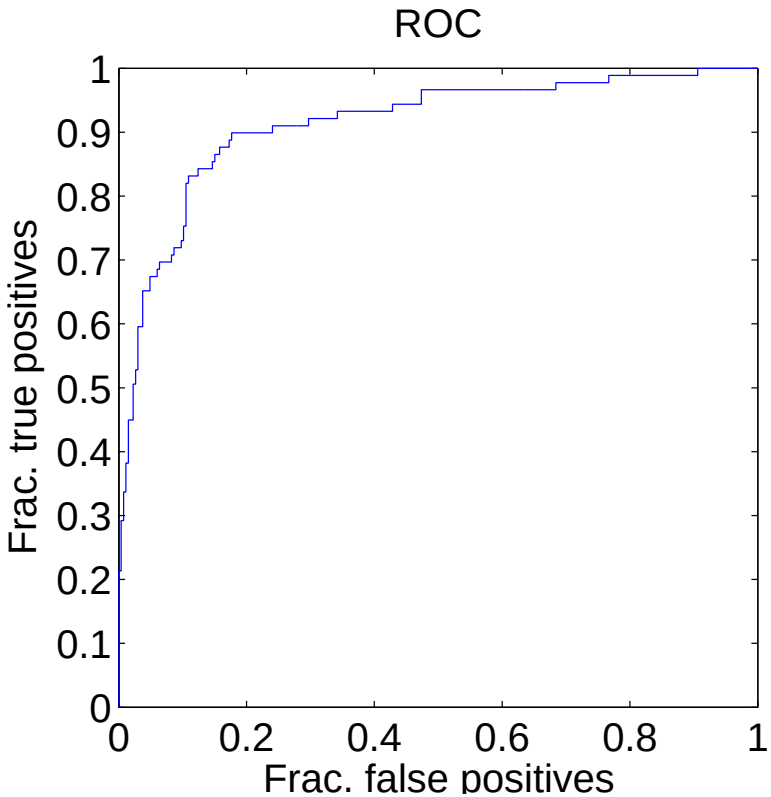
- Motivation: Determine importance of amino acid positions with respect to genotype-phenotype prediction accuracy
- Same idea as the sensitivity analysis for regression, but:
  - decrease in AUC (area under curve in ROC plot) instead of increase of MSE
  - random permutation of amino acids for each position

Application to HIV Receptor Interaction

- Data set of 355 samples with 63 AA positions
- Binary classification problem with 89 sequences that can use the CXCR4 receptor and 266 negatives
- Data set must be aligned first
- Ensemble of SVM, linear and k-NN classifiers
- Drawback: Quality of sensitivity analysis strongly depends on OOT prediction accuracy
- Pro: Method can be used universally for genotype-phenotype matching and other classification settings

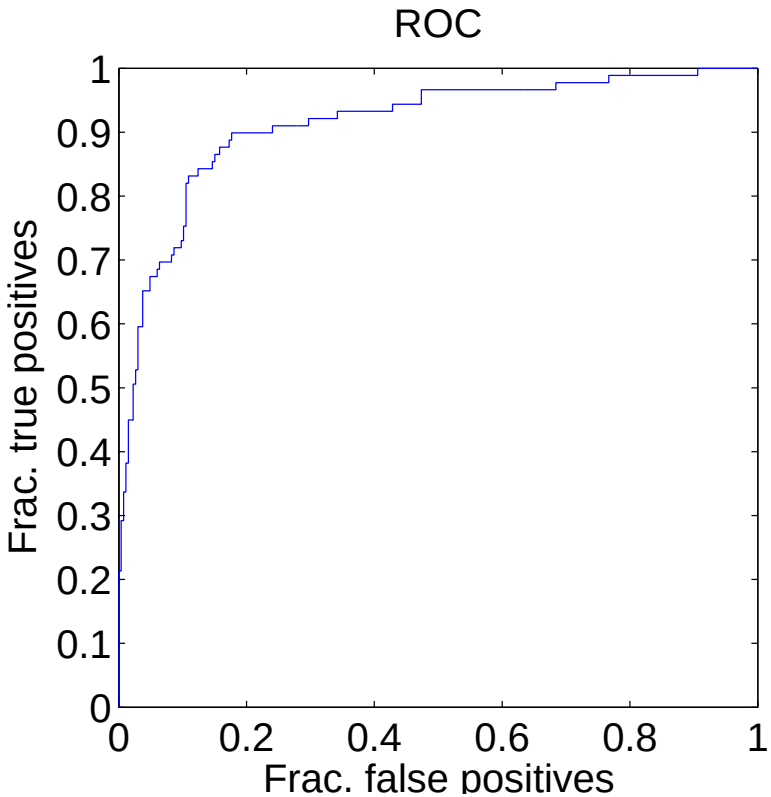
Sequence Analysis cont.

- Reasonable prediction accuracy on original data
- OOT AUC of 0.91

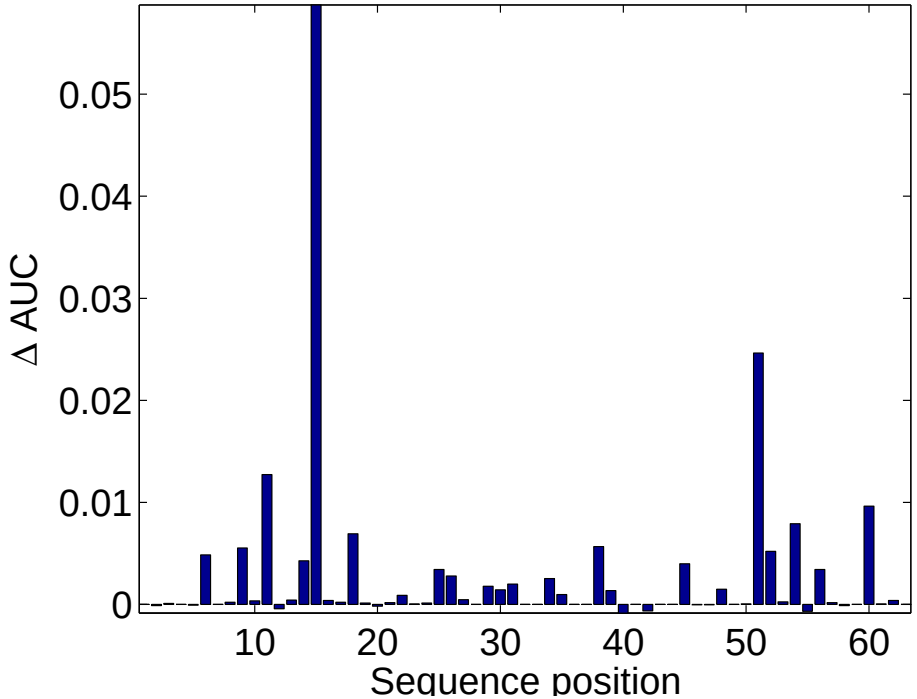


Sequence Analysis cont.

- Reasonable prediction accuracy on original data
- OOT AUC of 0.91



- Only a few sequence positions seem to be relevant:

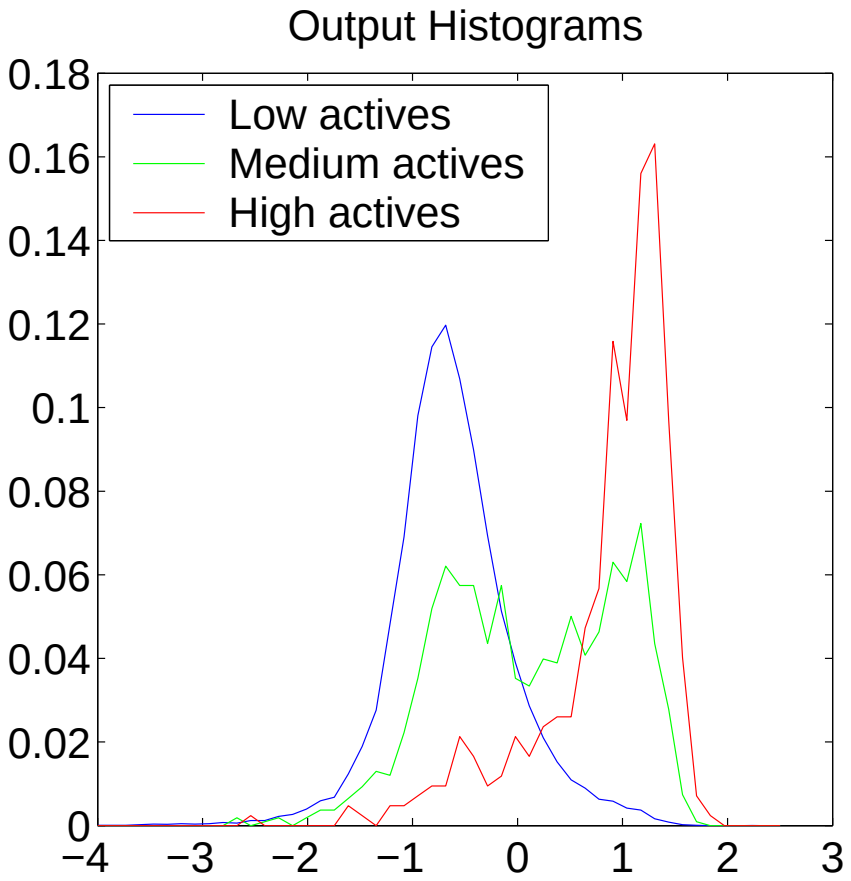


NCI Data Set

- DTP AIDS Antiviral Screen
- Total 42682 compounds (7 outtakes)
- Three classes:
  1. CA - Confirmed active 423
  2. CM - Confirmed moderate 1080 compounds
  3. CI - Confirmed inactive compounds
- No information about targets
- Random partition into training set of 35000 compounds and test set of 7682 compounds
- Ensemble of networks trained with classification loss

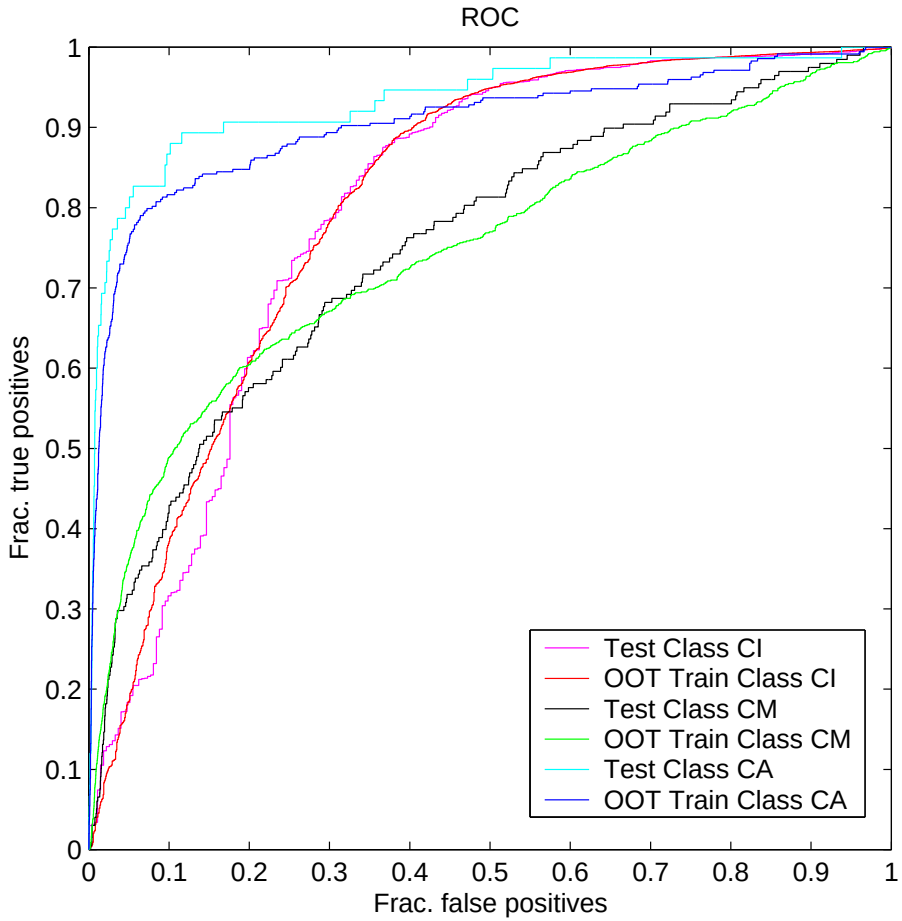
NCI Data Set

- DTP AIDS Antiviral Screen
- Total 42682 compounds (7 outtakes)
- Three classes:
  1. CA - Confirmed active 423
  2. CM - Confirmed moderate 1080 compounds
  3. CI - Confirmed inactive compounds
- No information about targets
- Random partition into training set of 35000 compounds and test set of 7682 compounds
- Ensemble of networks trained with classification loss

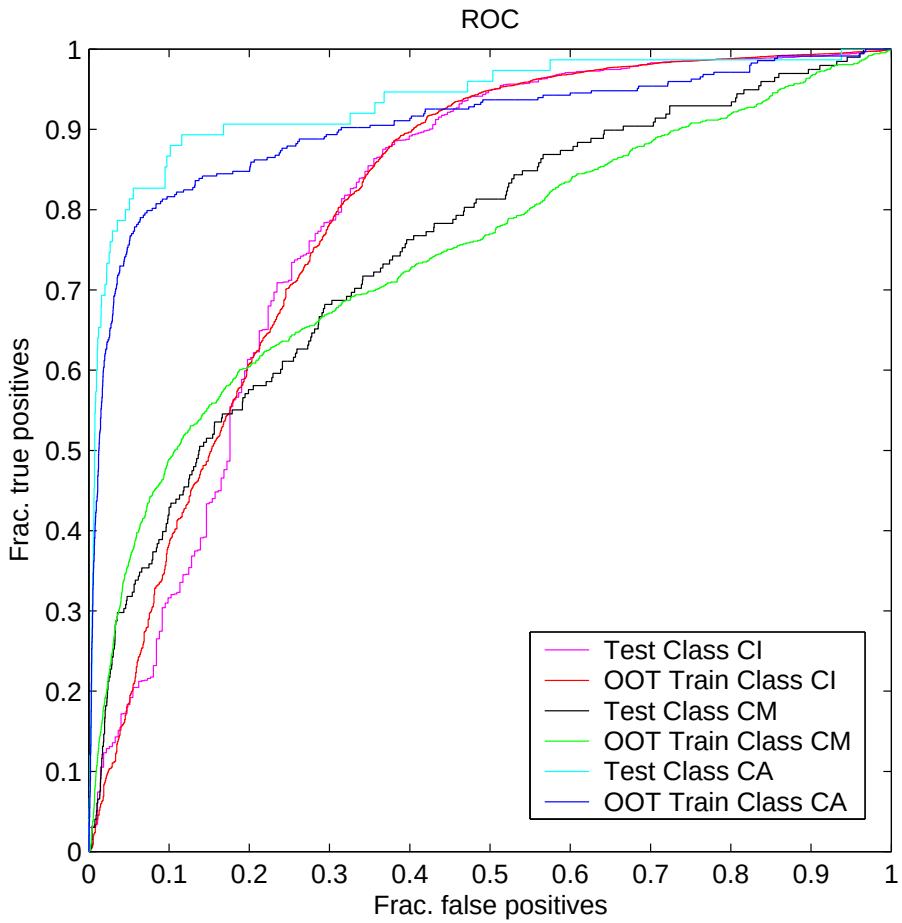


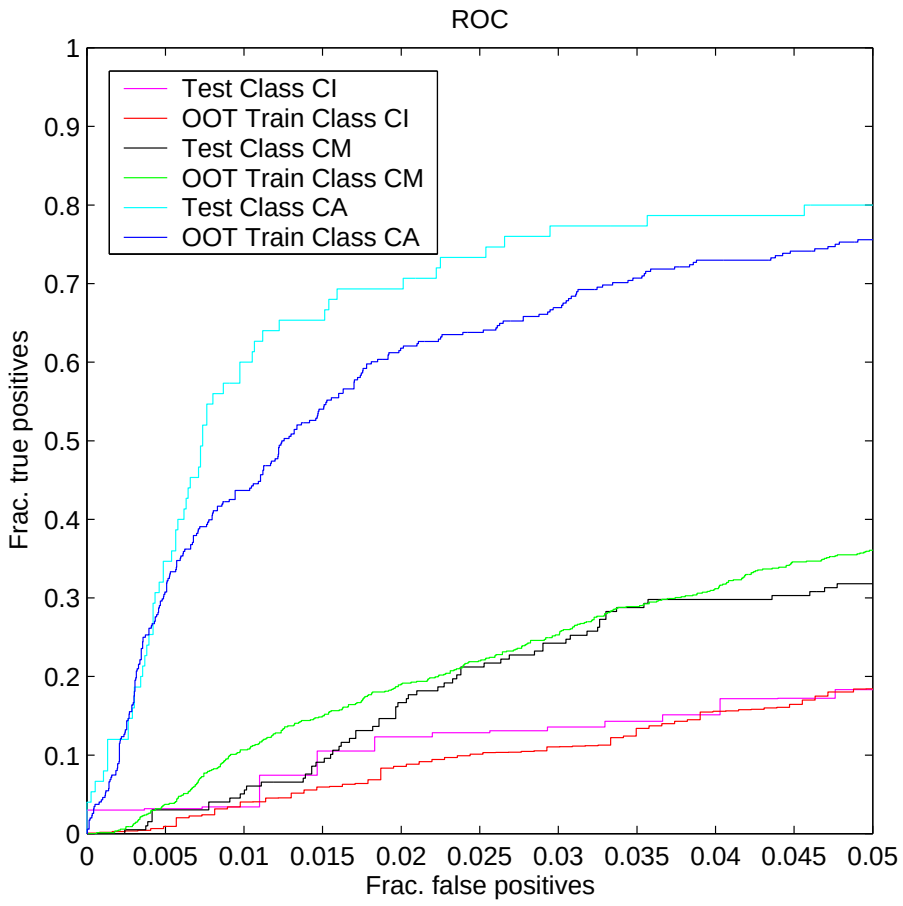
- Multiple modes of activity possible

Results : Classification on NCI Data Set



Results : Classification on NCI Data Set





Results : Toxicity Prediction

- EPA Fathead Minow Acute Toxicity Data Set of 617 industrial organic chemicals
- Predicting experimental LC 50
- MGN with 8 feature nets of 2-9 layers
- 50 fold Cross-Validation with 10% test on 577 compounds

$r^2 = 0.58$

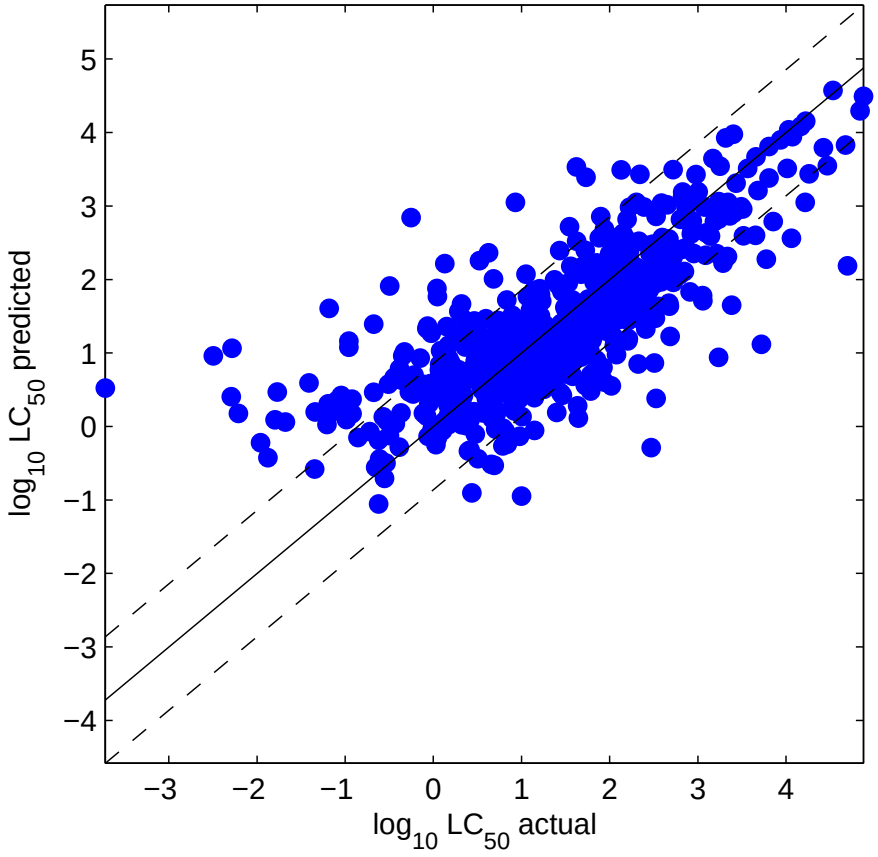
Russom, C.L., S.P. Bradbury, S.J. Broderius, D.E. Hammermeister, and R.A. Drummond (1997) Predicting modes of action from chemical structure: Acute toxicity in the fathead minnow (Pimephales promelas), Environmental Toxicology and Chemistry 16(5), 948-967

Results : Toxicity Prediction

- EPA Fathead Minow Acute Toxicity Data Set of 617 industrial organic chemicals
- Predicting experimental LC 50
- MGN with 8 feature nets of 2-9 layers
- 50 fold Cross-Validation with 10% test on 577 compounds

$r^2 = 0.58$

Russom, C.L., S.P. Bradbury, S.J. Broderius, D.E. Hammermeister, and R.A. Drummond (1997) Predicting modes of action from chemical structure: Acute toxicity in the fathead minnow (Pimephales promelas), Environmental Toxicology and Chemistry 16(5), 948-967

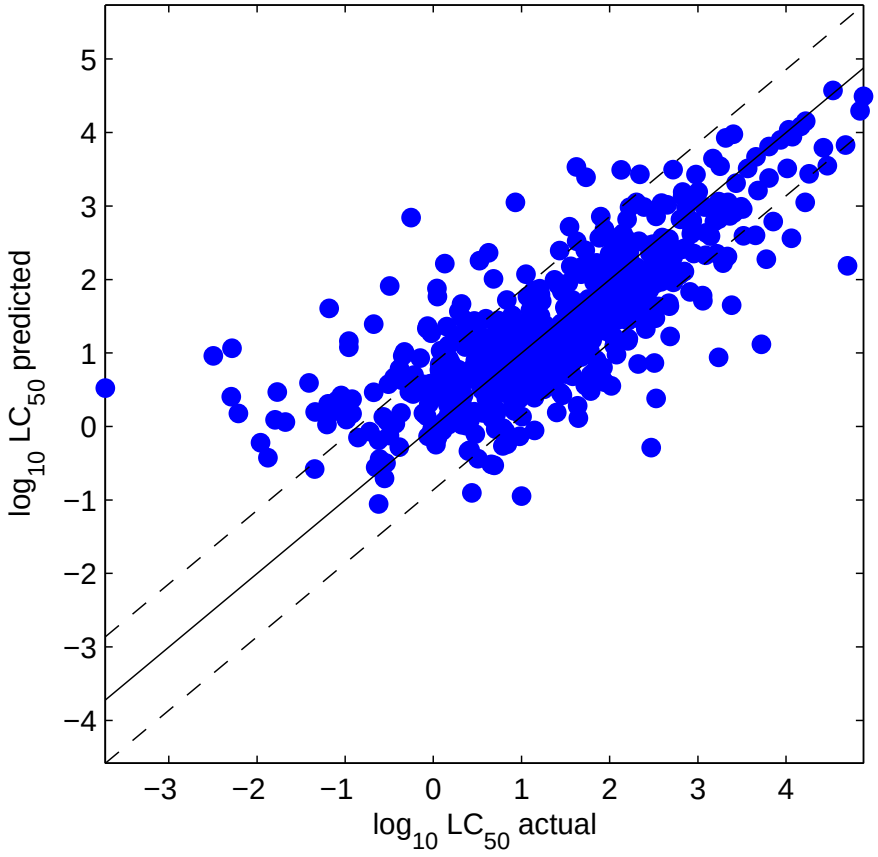


Results : Toxicity Prediction

- EPA Fathead Minow Acute Toxicity Data Set of 617 industrial organic chemicals
- Predicting experimental LC 50
- MGN with 8 feature nets of 2-9 layers
- 50 fold Cross-Validation with 10% test on 577 compounds

$r^2 = 0.58$

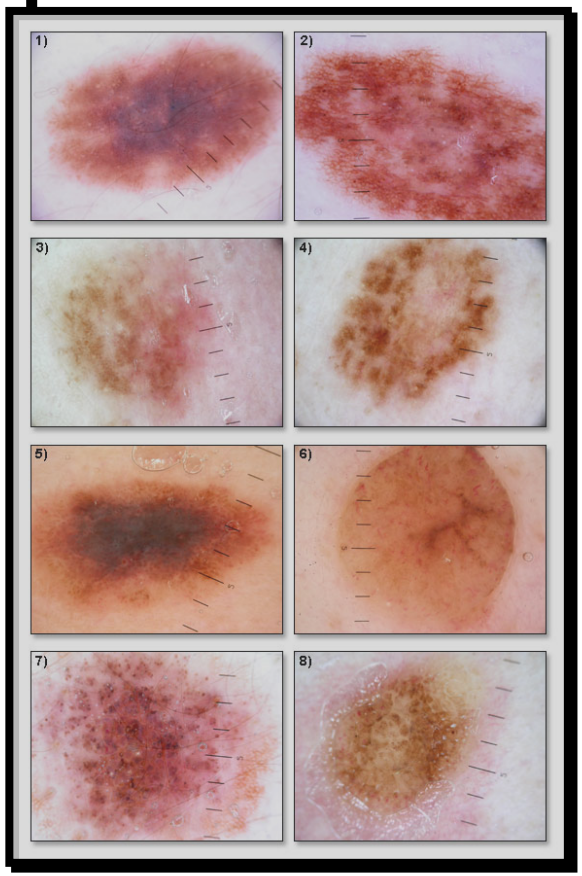
Russom, C.L., S.P. Bradbury, S.J. Broderius, D.E. Hammermeister, and R.A. Drummond (1997) Predicting modes of action from chemical structure: Acute toxicity in the fathead minnow (Pimephales promelas), Environmental Toxicology and Chemistry 16(5), 948-967



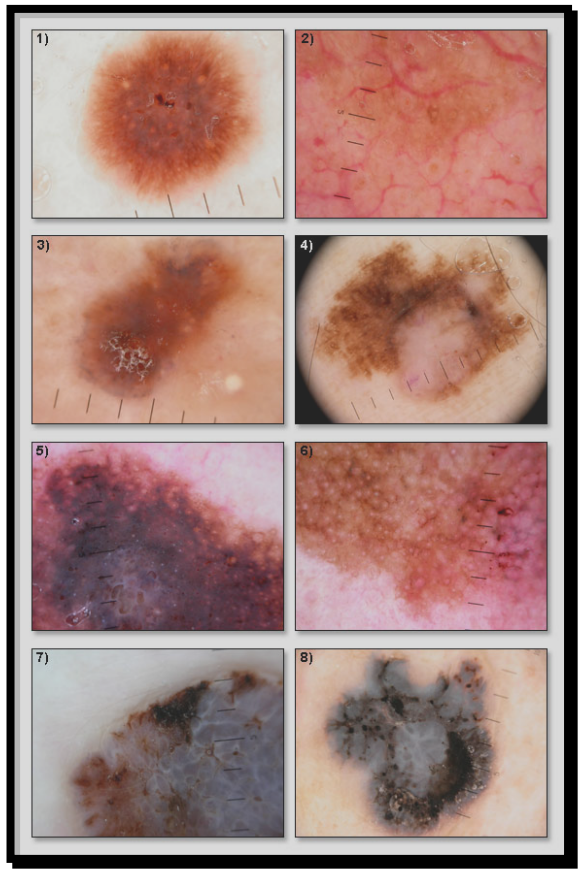
- Predictive Toxicity remains difficult

Image differentiation

Dysplastic



Melanoma



Measurements

Geometry:

- Vertical and horizontal symmetry
- Color symmetry
- Heigth and width
- Area of the lesion against the size of the photograph
- Perimeter (langth of borders)

Statistical Measurements:

- Color distribution (white, black and grey-blue),
- Estimated area
- Estimated perimeter
- Average distribution of RGB components in the lesion
- Average distribution of color components (**HSV, YIQ, YCbCr**)
- Binary connetions of color components

TDS (**T**otal **D**ermoscopy **S**core)

TDS = A * 1,3 + B * 0,1 + C * 0,5 + D * 0,5

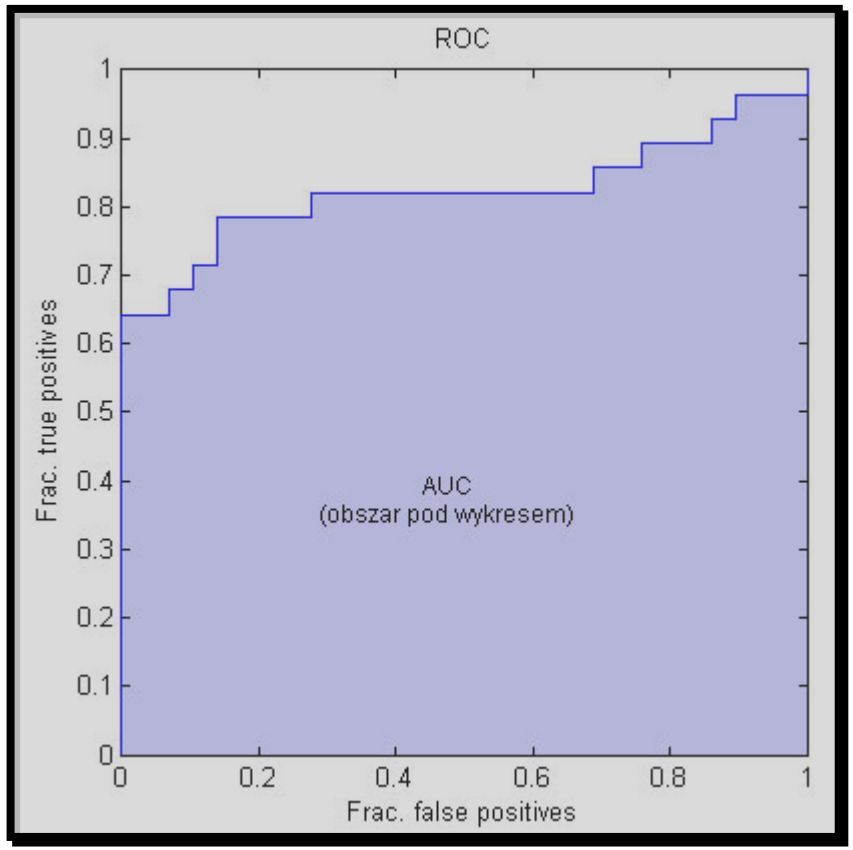
ABCD evaluation	
Property	TDS
Asymmetry **B**order **C**olor **D**ifferent structural components	x 1.3 x 0.1 x 0.5 x 0.5
outcome	**< 4.75** - benignant **4.8 - 5.45** – suspected melanoma **> 5.45 –** probable melanoma

Test for all coefficients

- AUC - **0.8239**

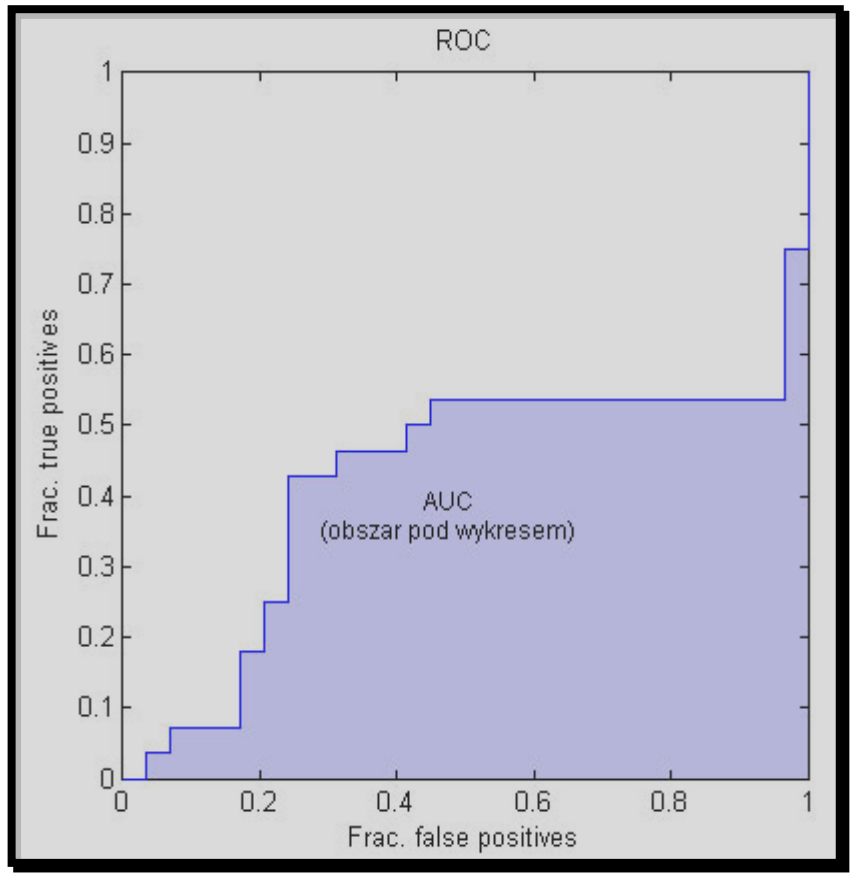
	No.	Coefficient
1	21	Sum of bckgrnd color comp
2	33	Average Cr component
3	39	Average V of background
4	14	Average red
5	15	Average green
6	38	Average S of background
7	29	Average S
8	16	Average blue
9	31	Average luminance
10	36	Average Q
11	45	Average Q of background
12	4	Estimated size (px)
13	34	Average Y
14	13	symmmetry (%)
15	1	Area of the lesion (%)
16	2	Area of the lesion (px)
17	11	Gray-blue (px)
18	44	Average I of background
19	3	Area of background (px)
20	5	Height (px)
21	35	Average I
22	37	Average H of background
23	43	Average Y of background

	No	Coefficient
24	9	White color (px)
25	30	Average V
26	42	Average comp. Cr bckgrnd
27	40	Average luminance bckgrnd
28	7	Borders
29	32	Average comp. Cb
30	18	Average green in bckgrnd
31	19	Average blue in bckgrnd
32	6	Width (px)
33	28	Average H
34	10	Black color (px)
35	17	Average red in backgrnd
36	12	Grey-blue
37	20	Sum of color components
38	25	Binary sum of GBR
39	41	Average Cb comp
40	8	backgrnd
41	22	Estimated borderline
42	24	Binary RGB composition
43	23	Binary GRB composition
44	27	Binary RBG composition
45	26	Binary BGR composition



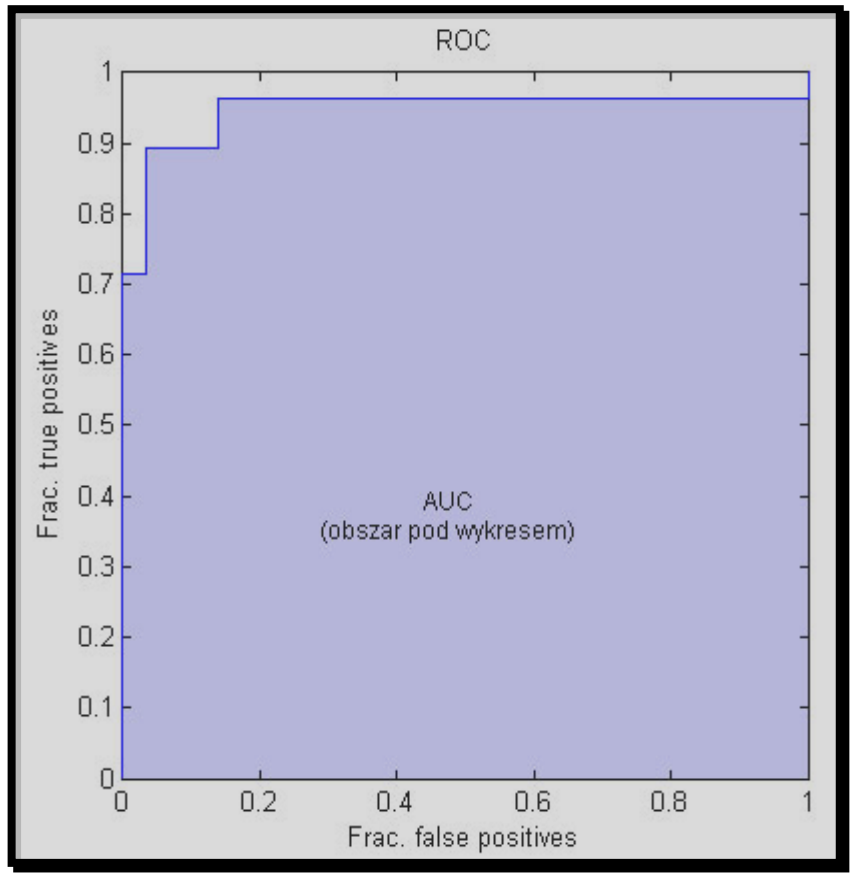
Single coefficient test

- AUC - **0.4212**



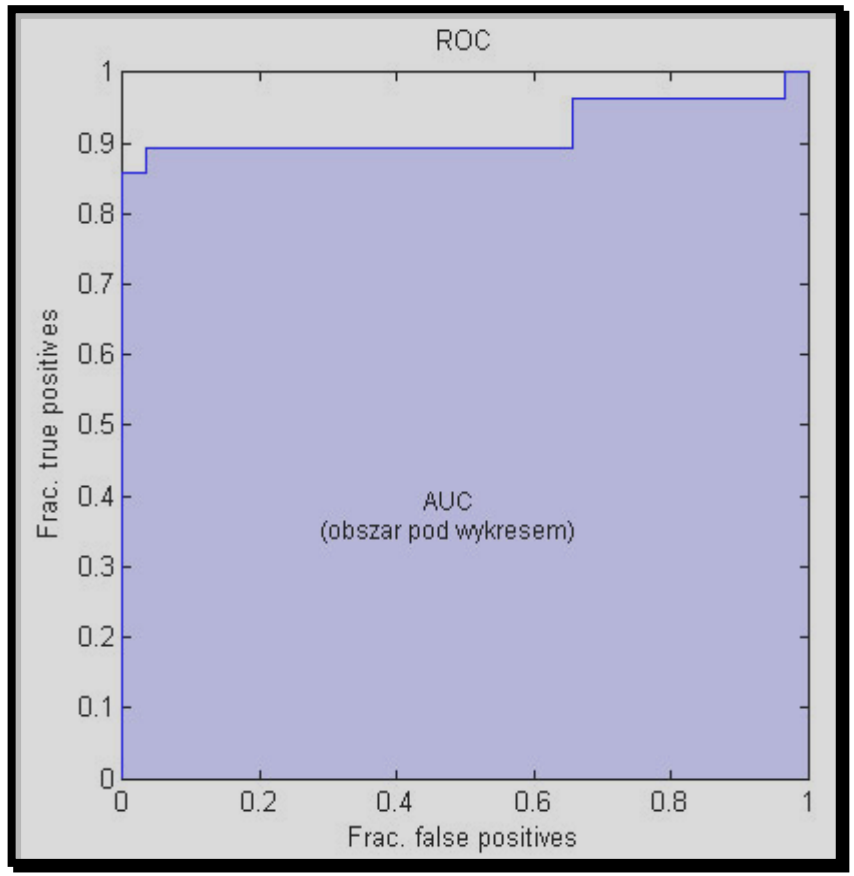
6 coefficients

- AUC - **0.9483**



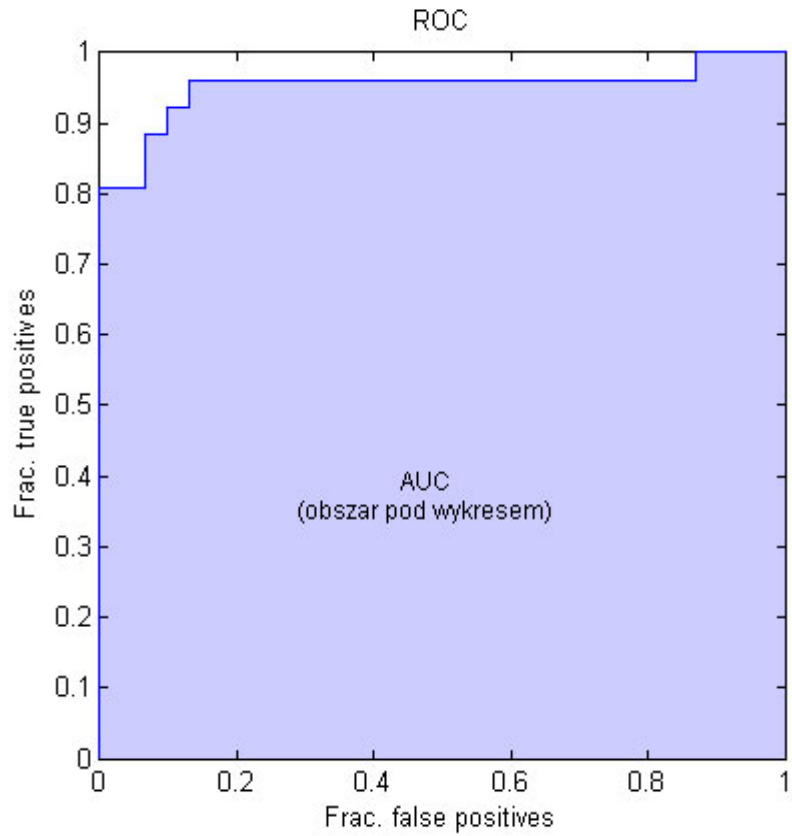
Test with 15 strongest coefficients

- AUC - **0.9175**



Test for whole data set

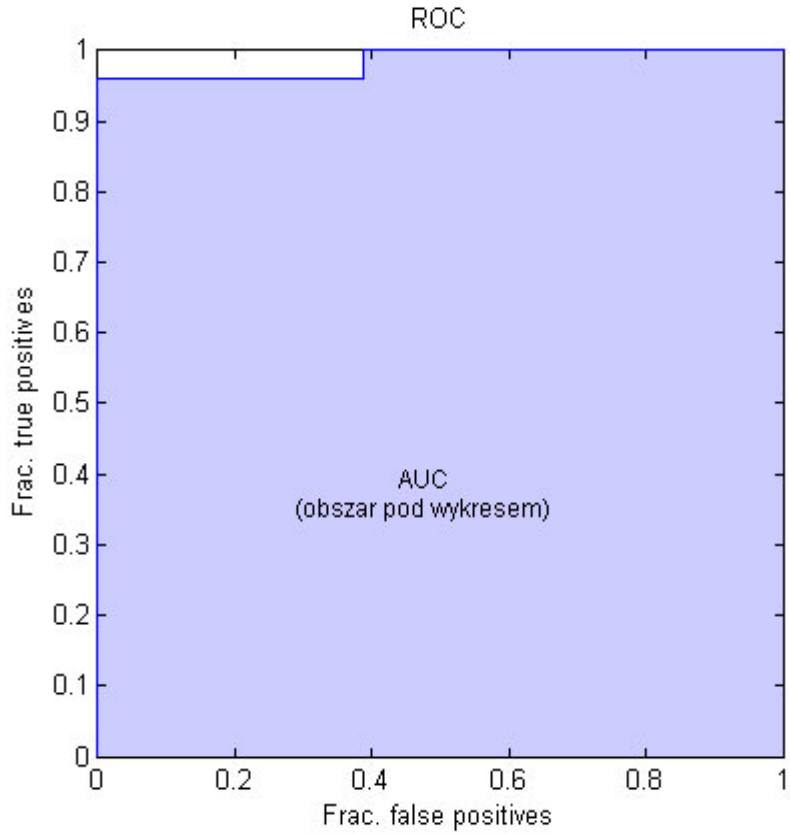
- AUC - **0.9529**



Test for 15 best coefficients

- AUC - **0.9851**

	No.	Coefficient
1	21	Sum of color comp of bckgrnd
2	14	Average red
3	15	Average green
4	13	symmetry (%)
5	16	Average blue
6	34	Average Y
7	38	Average S of background
8	43	Average Y of background
9	12	Grey-blue – black and white
10	36	Average Q
11	18	Average green of background
12	31	Average luminance
13	39	Average V of background
14	45	Average Q of background
15	30	Average V



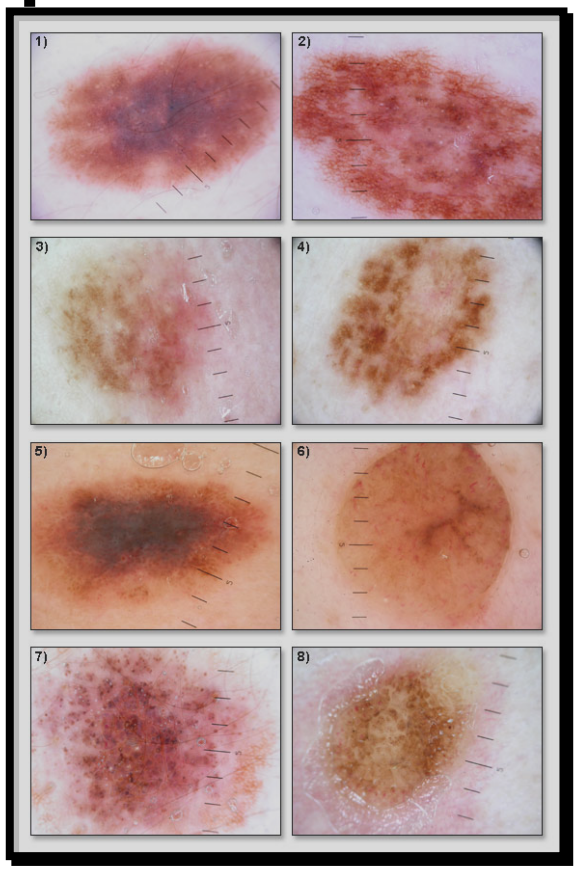
Set of 17 coefficients (best results)

	No	Coefficient
1	21	Sum of color comp. of the background
2	14	Average red
3	15	Average green
4	13	symmetry (%)
5	16	Average blue
6	34	Average Y
7	38	Average S of the background
8	43	Average Y of the background
9	12	Grey-blue – black and white
10	36	Average Q
11	18	Average green of the background
12	31	Average luminance
13	39	Average V of the background
14	45	Average Q of the background
15	30	Average V
16	35	Average I
17	44	Average I of the background

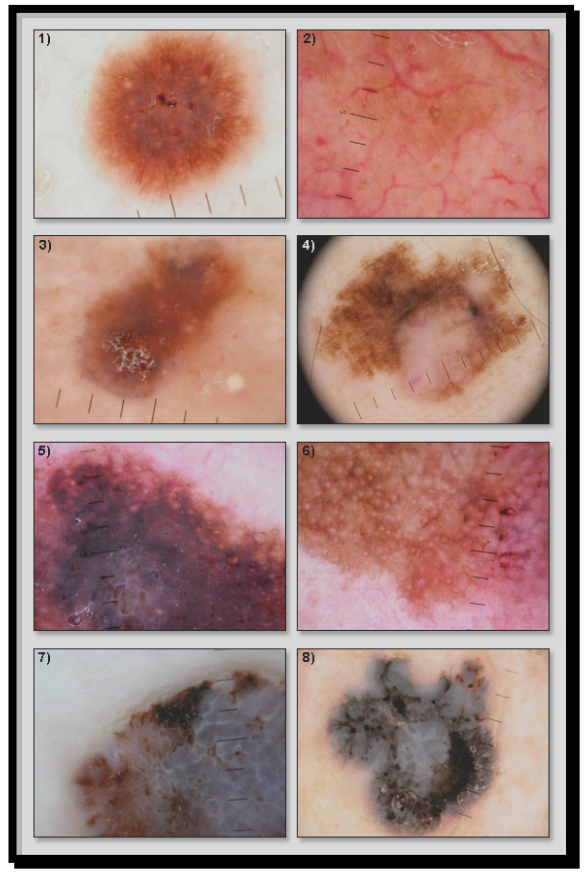
- AUC – 0.**9963**

Image verification

Dysplastic



Melanoma



Summary

- Ensemble methods for classification and regression
- ENTOOL - Ensemble toolbox for Matlab
- State-of-the art machine learning techniques
- Variety of primary and secondary model types
- Out-of-Train technique for accessing generalization error
- Sensitivity Analysis for classification and regression
- Application to skin permeability
- Application to genotype-phenotype matching

Summary

- Ensemble methods for classification and regression
- ENTOOL - Ensemble toolbox for Matlab
- State-of-the art machine learning techniques
- Variety of primary and secondary model types
- Out-of-Train technique for accessing generalization error
- Sensitivity Analysis for classification and regression
- Application to skin permeability
- Application to genotype-phenotype matching
- Applicable to data sets of any size
- Classification of active/inactive compounds NCI Antiviral Screen
- Toxicity prediction as regression problem

Literature

- Krogh, Vedelsby Neural Network Ensembles, Cross Validation and Active Learning Advances in Neural Information Processing Systems 7, MIT Press 1995
- Peronne, Cooper When networks disagree: Ensemble methods for neural networks Neural Networks for Speech and Image Processing, Chapman Hall 1993
- Hastie, Tibshirani, Friedman The Elements of Statistical Learning Springer 2001
- Vapnik The Nature of Statistical Learning Theory Springer 1999
- Chih-Chung Chang, Chih-Jen Lin LIBSVM : a library for support vector machines Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`, 2001
- Freund Short introduction to boosting 1999
- Sacks, Welch, Mitchell, Wynn Design and analysis of computer experiments Statistical Science, 4(4):409-435, 1989
- Domingos A unified bias-variance decomposition for zero-one and squared loss Proceedings of the Seventeenth National Conference on Artificial Intelligence , 2000
- Breiman Bagging Predictors . Machine Learning, 24, 1996