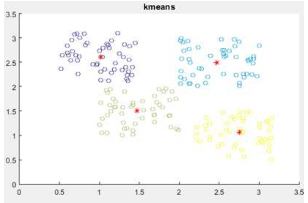
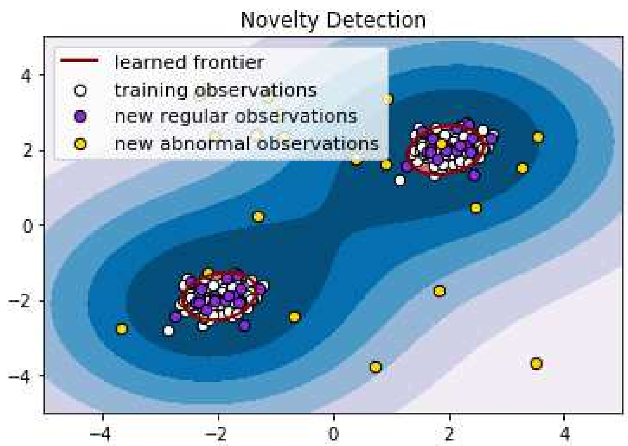
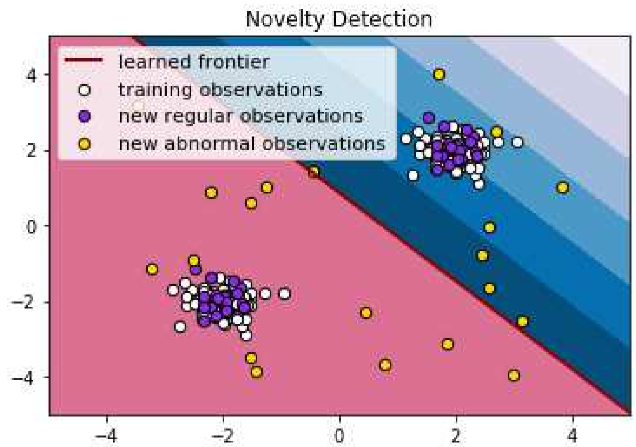
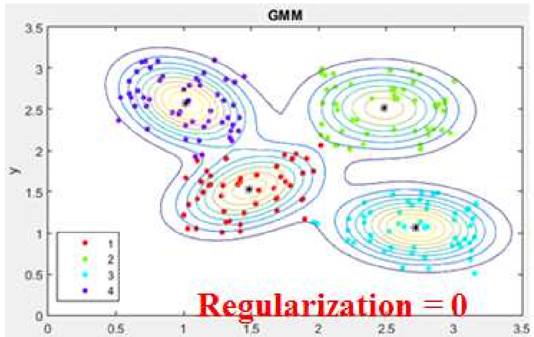


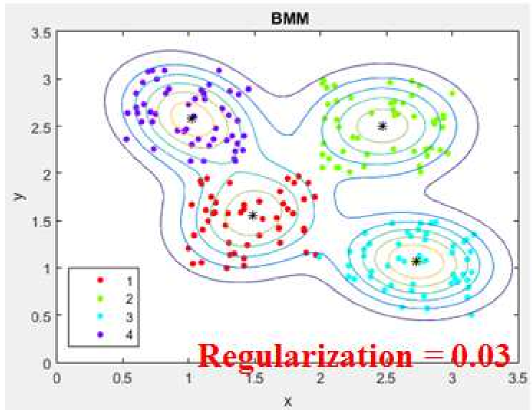
SVM One Class Classification

- **Matlab fitcsvm( )** **or** **Sklearn svm.OneClassSVM( )**
  - An unsupervised method to learn a decision function for novelty detection.
  - Classify new data as how similar or how different to the training set. ***k*NN??**
  - **Like clustering???** Compare either to *k*-means or ***GMM*** (Gaussian Mixture Model).
  - http://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html#sklearn.svm.OneClassSVM
  - http://scikit-learn.org/stable/auto_examples/covariance/plot_outlier_detection.html#sphx-glr-auto-examples-covariance-plot-outlier-detection-py
  - http://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html







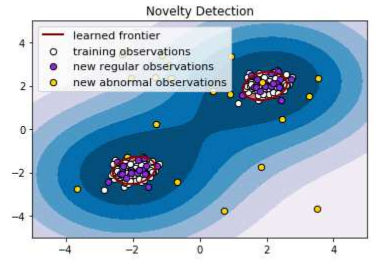



```
# kernel must be one of 'linear', 'poly', 'rbf', 'sigmoid'
clf = svm.OneClassSVM(nu=0.1, kernel="rbf", gamma=0.1)
clf.fit(X_train);        y_pred_test = clf.predict(X_test)
```

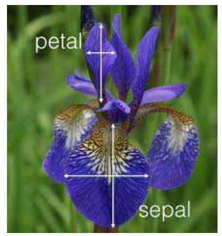
One-Class SVM Experiments

```
% one-class SVM
yy = Y * 0;        %% make all labels = 0
mdl_c1 = fitcsvm(X, yy, , 'KernelFunction', 'rbf');
[~, scores_c1] = predict(mdl_c1, X);
```

- sklearn dataset
  - Build **only** 1 **one**-class SVM from …
  - Data of (**one** class (majority) + outliers (minority).

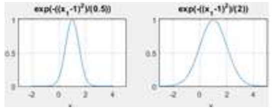


- Iris dataset
  - Build 2 **one**-class SVMs from …
  - Unbalanced data of (**two** classes).
  - Compare to 1 **two**-class SVM.

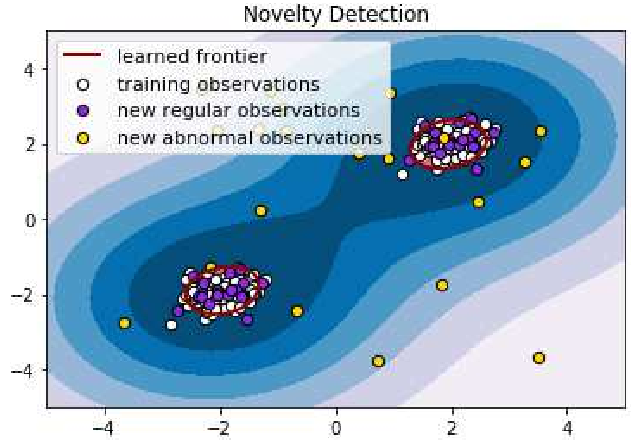


- Ovarian cancer dataset
  - Build 2 one-class SVMs from …
  - **Super** unbalanced data of (**two** classes).
  - **4,000**-dimension data.

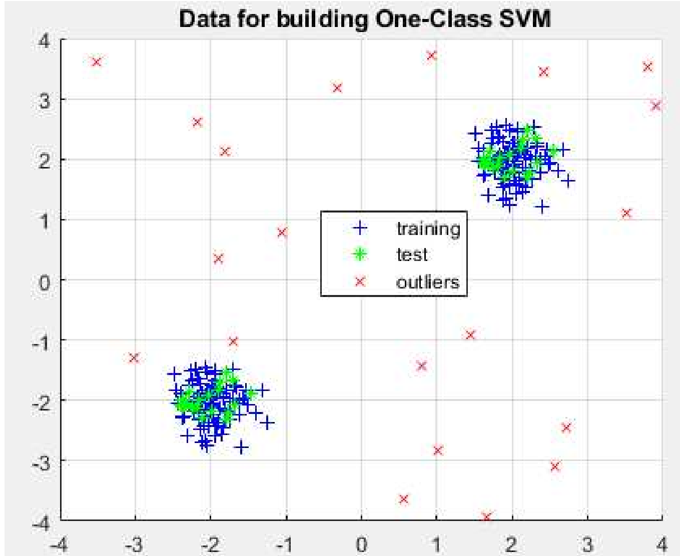
SVM RBF One Class, NO Kernel Scale

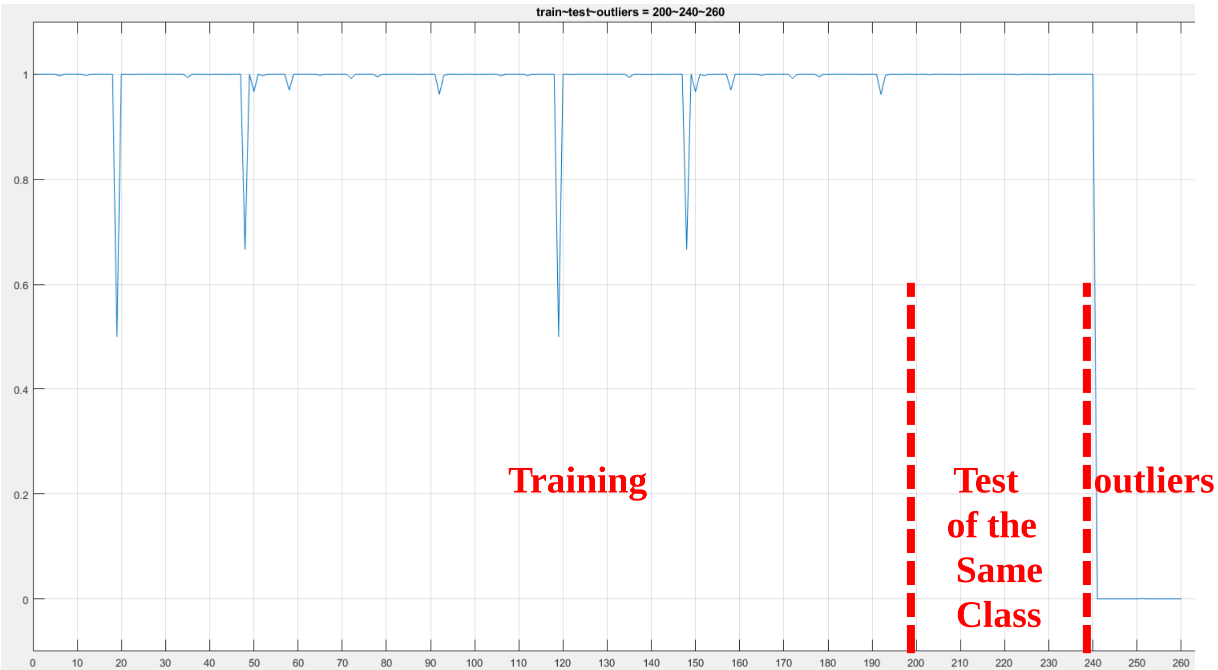


- http://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html#sphx-glr-auto-examples-svm-plot-oneclass-py

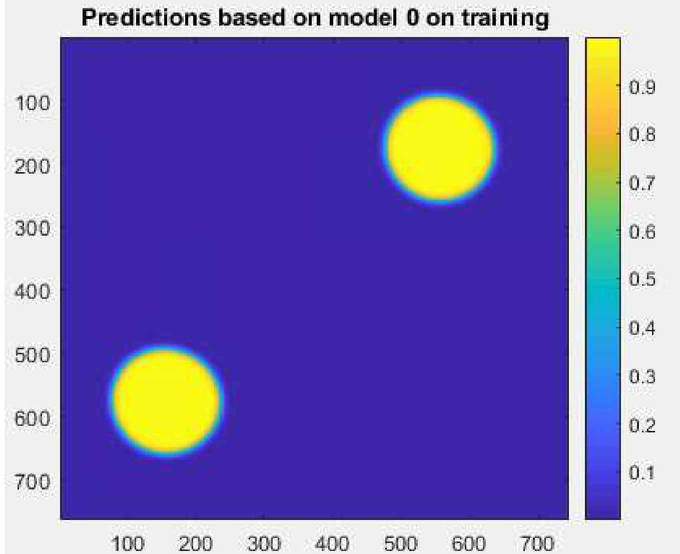



```
# data
rng(10)

XX = 0.3 * randn(100, 2);
X = [XX + 2; XX - 2];
Y = zeros(size(X, 1), 1);

XX = 0.3 * randn(20, 2);
X_test = [XX + 2; XX - 2];

% Generate outliers
a = -4;    b = 4;
X_outliers = (b-a).*rand(20,2) + a;
```



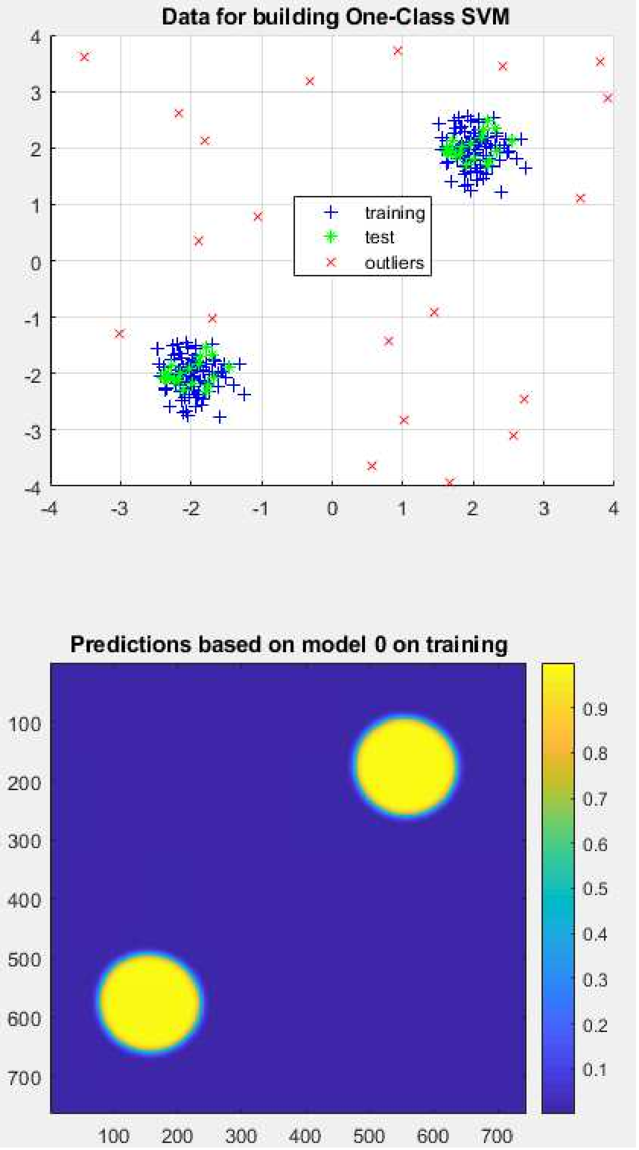


Probability

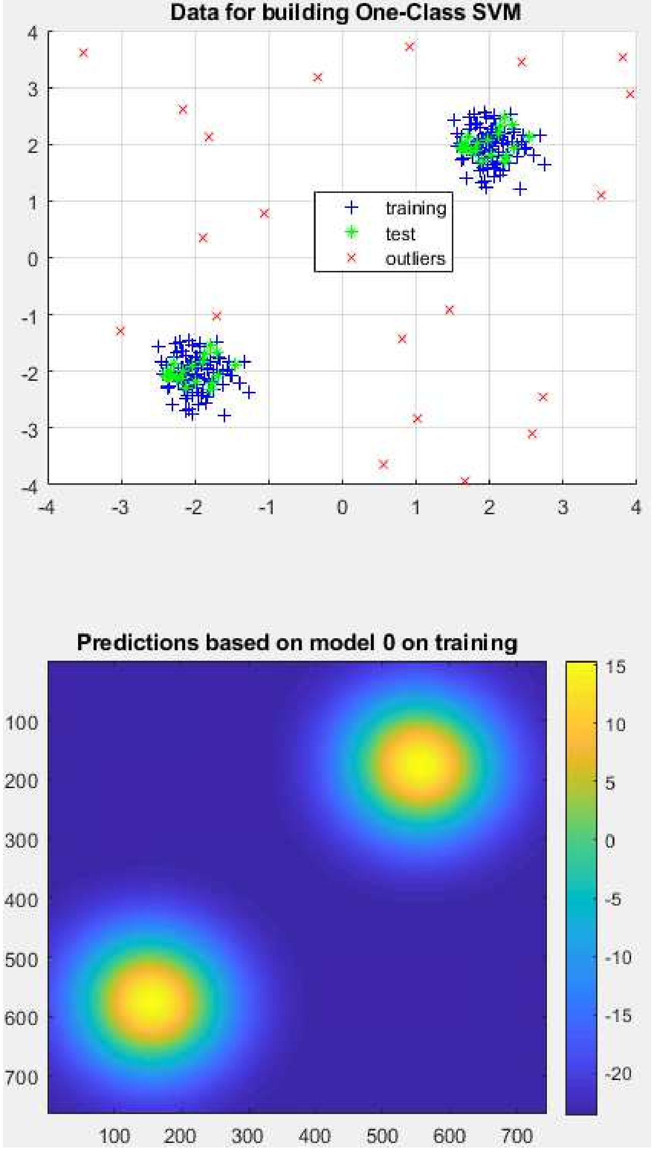


SVM RBF One Class, **NO** Kernel Scale

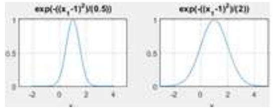
Probability



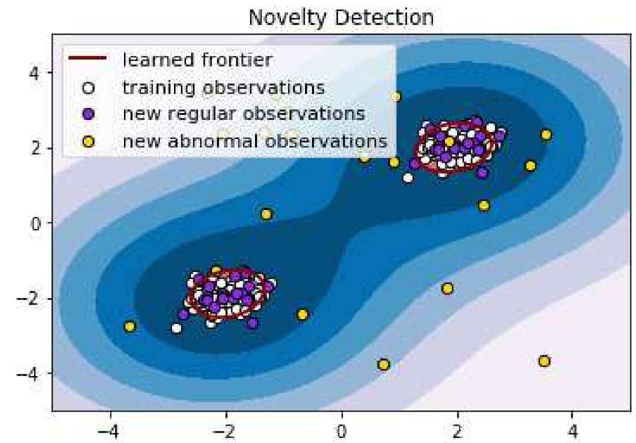
$W^{T}X$

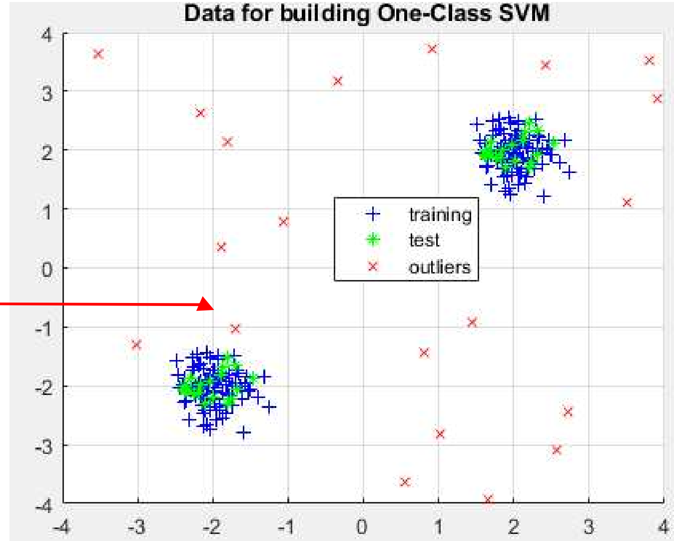


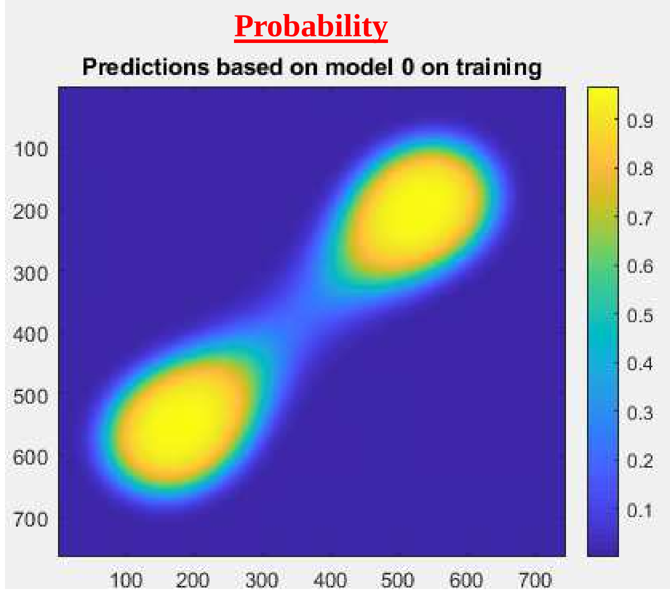
SVM RBF One Class, **Kernel Scale = 3.5**

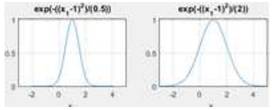


- http://scikit-learn.org/stable/auto_examples/svm/plot_oneclass.html#sphx-glr-auto-examples-svm-plot-oneclass-py



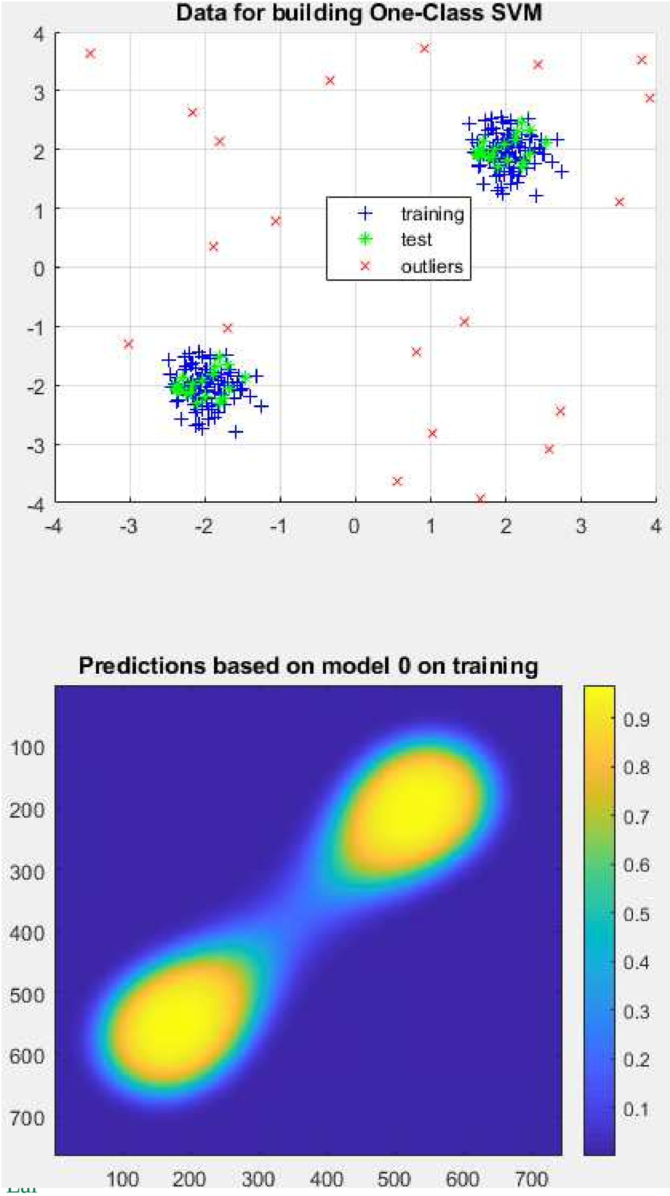




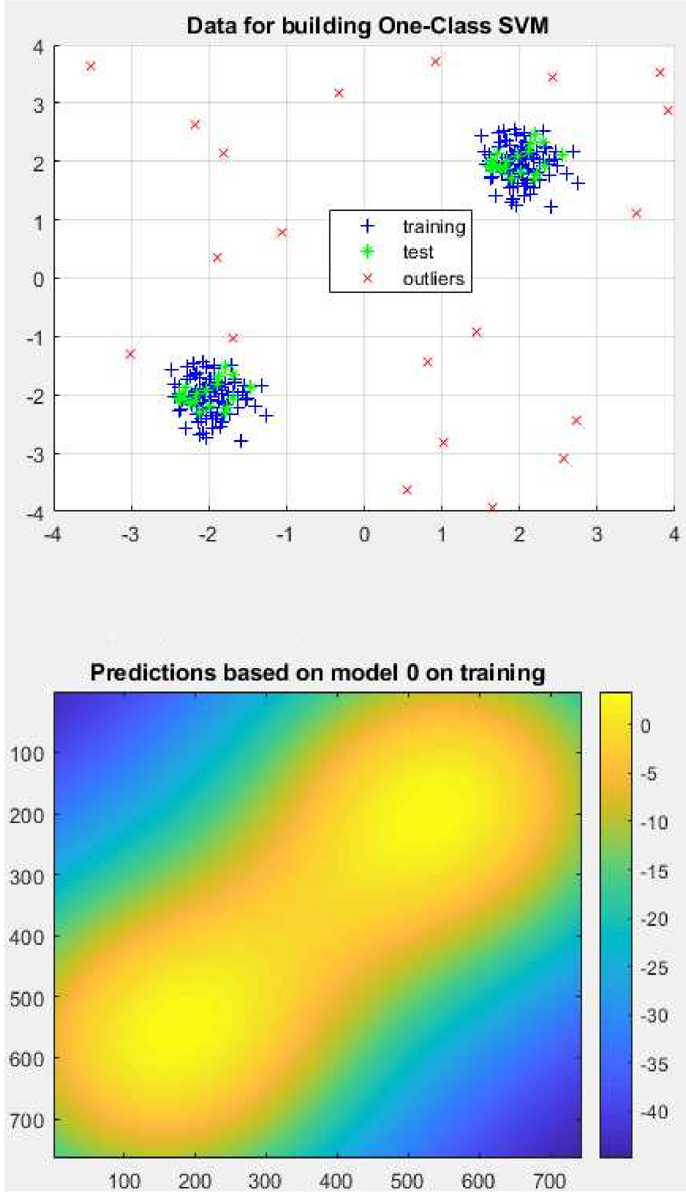


SVM RBF One Class, **Kernel Scale = 3.5**

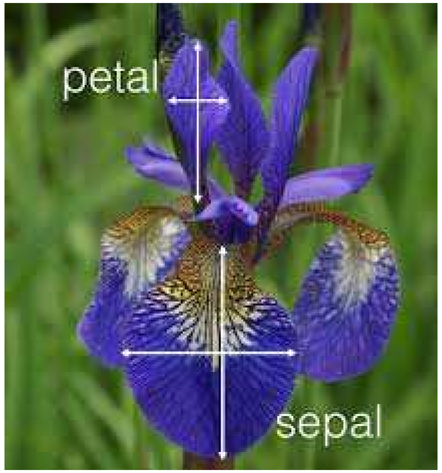
Probability



$W^T X$



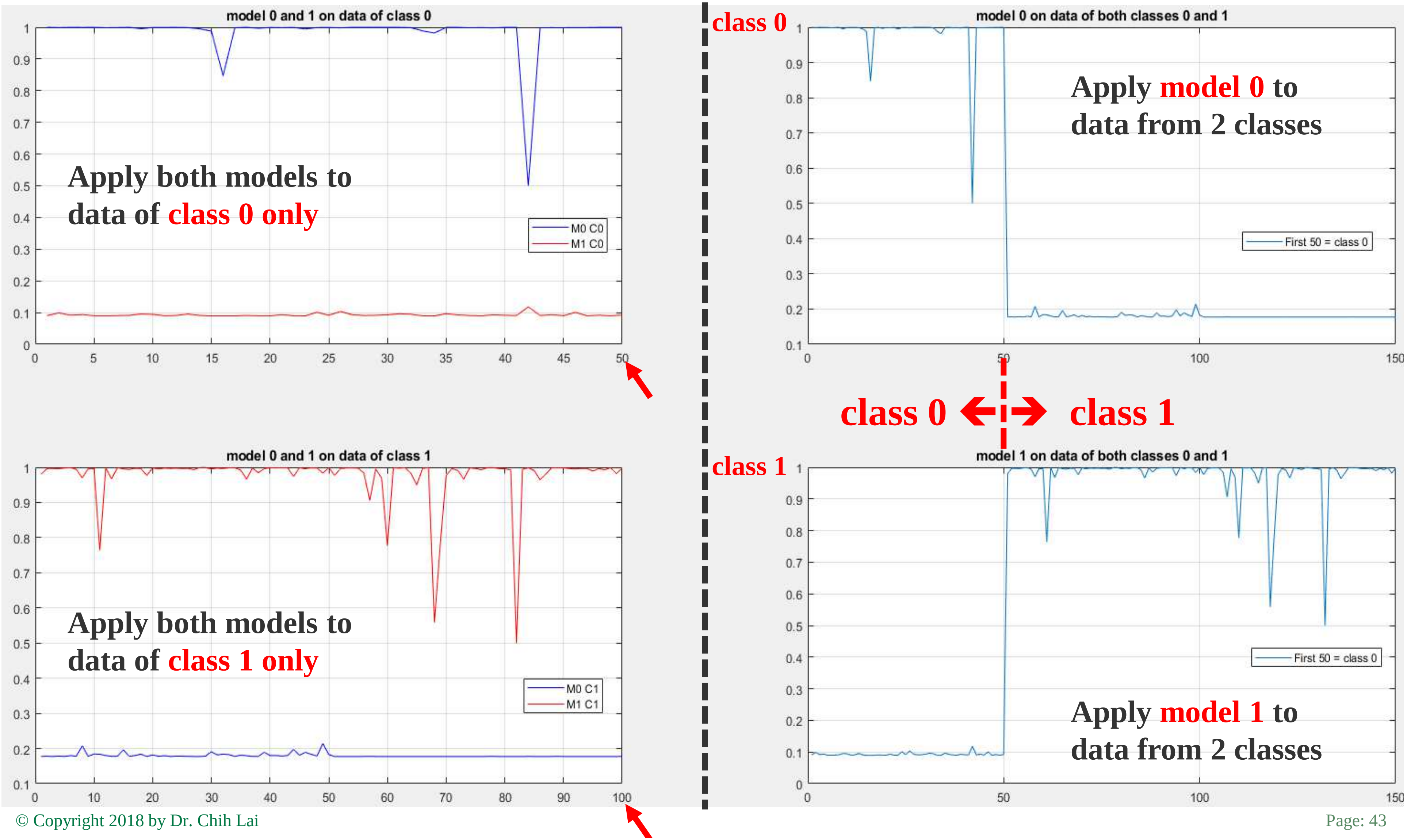
Iris Dataset



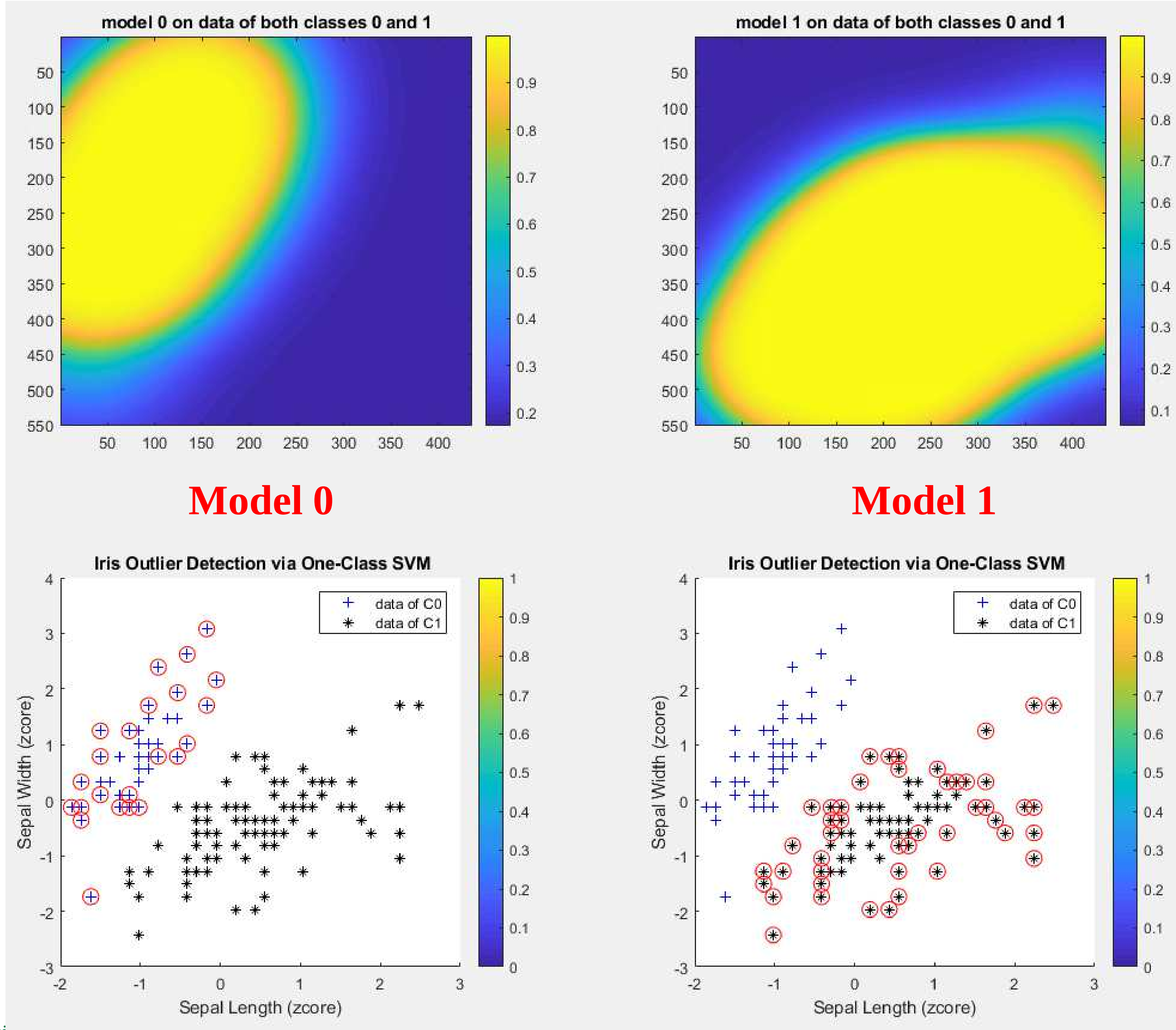

```
idx1 = find(Y);      idx0 = find(Y == 0);
x0 = X(idx0,:);      y0 = Y(idx0);
x1 = X(idx1,:);      y1 = Y(idx1);
% for class 0
mdl_c0 = fitcsvm(x0, y0, , 'KernelFunction', 'rbf');
[~, scores_c0] = predict(mdl_c0, x0);

% for class 1 …
```

SVM / RBF One Class for **Unbalanced** Iris (50 vs. 100) Dataset

- Build model 0 (M0) from data of class 0 (first 50 rows) only.
- Build model 1 (M1) from data of class 1 (100 rows) only.



SVM / RBF One Class for **Unbalanced** Iris (50 vs. 100) Dataset– Cont'd



SVM / RBF for **Unbalanced** Iris (50 vs. 100) Data– One 2-Class SVM

- Build **ONE** 2-Class model from both classes.

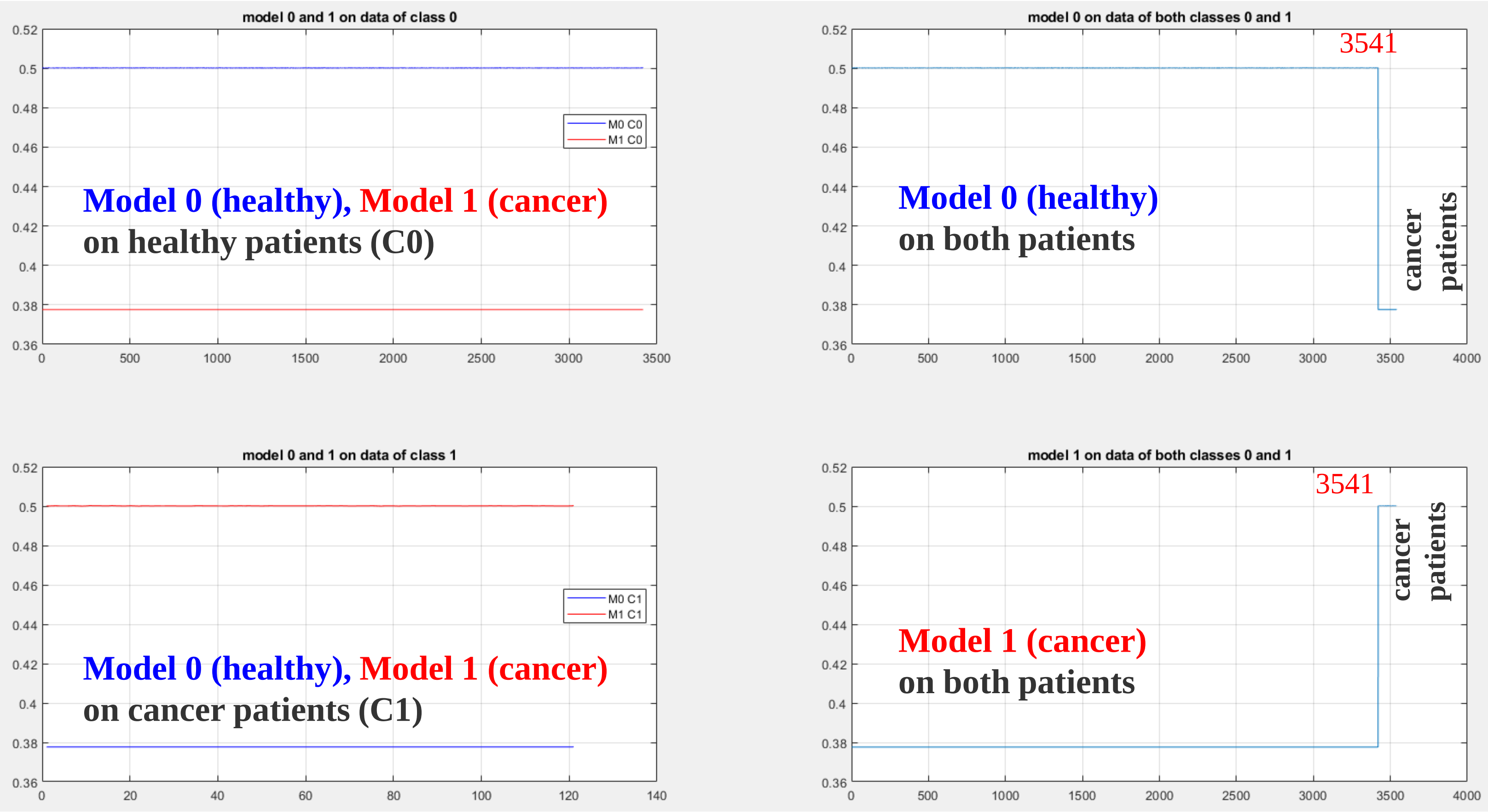
Ovarian Cancer Dataset

Detecting Super Unbalanced Ovarian Cancer Patients (One-Class)

- M0 was trained on healthy patients (C0).
- M1 was trained on ovarian cancer patients (C1).

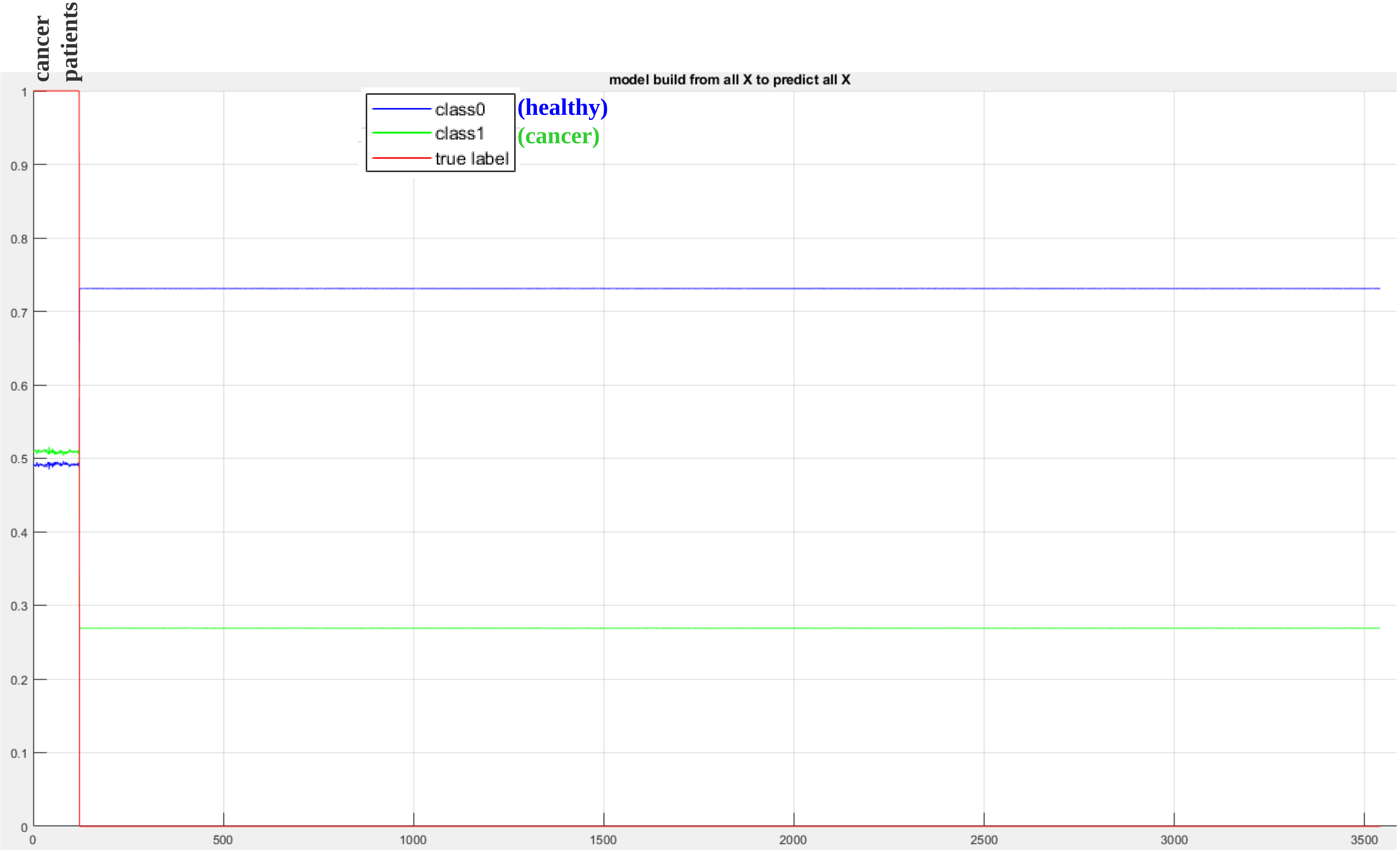
```
idx1 = find(Y);      idx0 = find(Y == 0);
x0 = X(idx0,:);      y0 = Y(idx0);
x1 = X(idx1,:);      y1 = Y(idx1);
mdl_c0 = fitcsvm(x0, y0, , 'KernelFunction', 'rbf');
[~, scores_c0] = predict(mdl_c0, x0);
```

- Duplicate healthy records couple times to create a skewed dataset
  - Original data, total 261, Cancer = 121 (46%).   Duplicate data, total 3662, Cancer = 121 (3.4%)



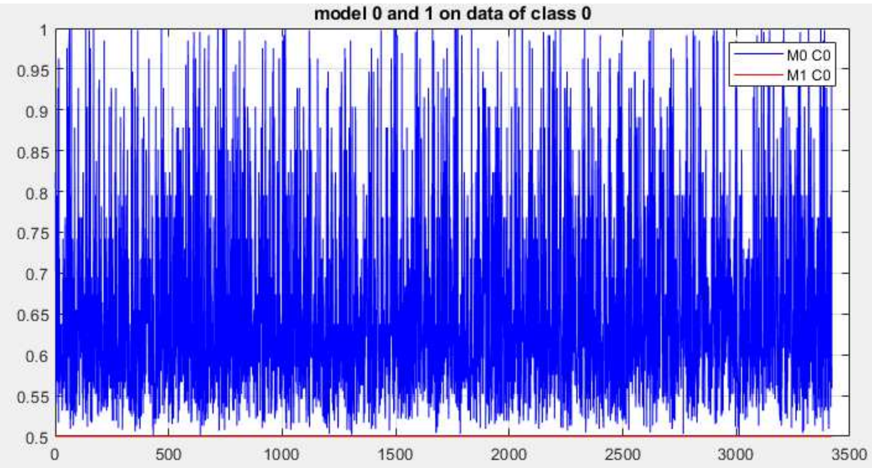
Build One 2-Class SVM

- Build **one** **2-class** SVM model from **ALL** (unbalanced) patients.

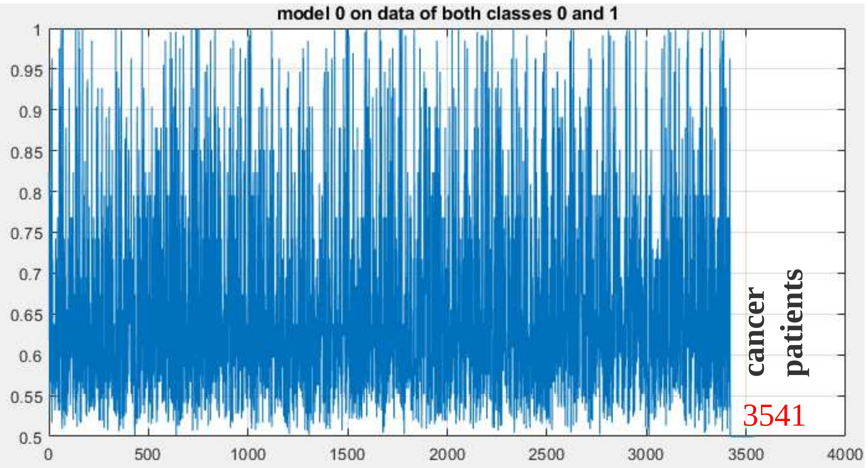


Super Unbalanced Ovarian Cancer Patients (1-Class), **Kernel Scale = 0.1**

Model 0 (healthy), Model 1 (cancer) on healthy patients

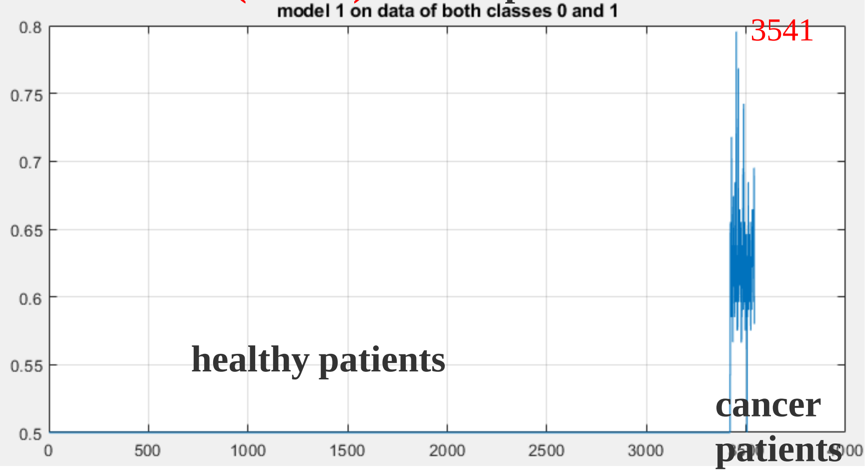


Model 0 (healthy) on both patients



Model 0 (healthy), Model 1 (cancer) on cancer patients

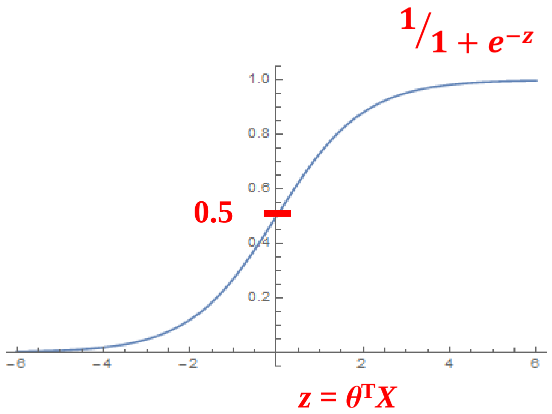
Model 1 (cancer) on both patients



Apply One-Class Classification Using Logit??

- Objective Function for Logistic Regression
  - <u>**minimize**</u> ***negative log likelihood***

$$\frac{-1}{m}\sum_{i=1}^{m}[Y_i log(P_i) + (1 - Y_i) log(1 - P_i)]$$



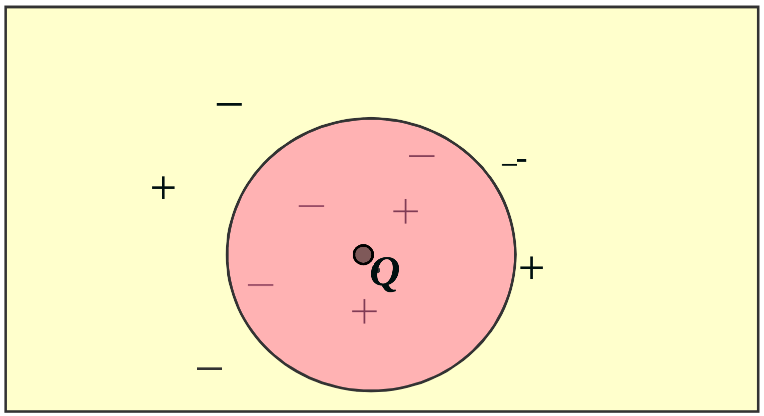
- Object function for SVM Kernel
  - Min $L_D = \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \boldsymbol{x_i x_j} - \sum_{i=1}^{m}\alpha_i$ **(Dual form)**
  - Min $L_D = \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \Phi(\boldsymbol{x_i})\Phi(\boldsymbol{x_j}) - \sum_{i=1}^{m}\alpha_i$

Instance-Based Methods

- **<u>Eager</u>** instance-based learning, like SVM-RBF.

- **<u>Lazy</u>** instance-based learning. Store past data, **<u>NO</u>** model construction.
  - Approach 1– ***k*-nearest neighbor (*k*NN)**
    - All instances (records) are represented as points in the *n*-D Euclidean space.
    - Assign the majority class of the nearest neighbors to the new (unseen) data.
    - For each query, finding *k*NN can be very time consuming.
    - *k*-nearest neighbors can be far away (very dissimilar) from *Q*.
  - Approach 2– *range query*

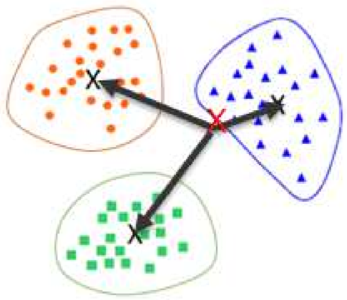
+ ***k*=11**



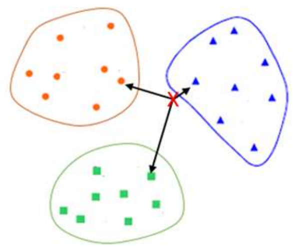
Outlook	Temp.	Humid	Windy	Play
S	W	H	F	N
S	W	H	T	N
O	W	H	F	Y
R	M	H	F	Y
R	C	L	F	Y
R	C	L	T	N
O	C	L	T	Y
S	M	H	F	N
S	C	L	F	Y
R	M	L	F	Y
S	M	L	T	Y
O	M	H	T	Y
O	W	L	F	Y
R	M	H	T	N

*k*NN for Digit Recognition

- Compute distance between each test instance against **ALL** training data
  - Predict query image based on majority of ***k***NN digits.   Slow to run.



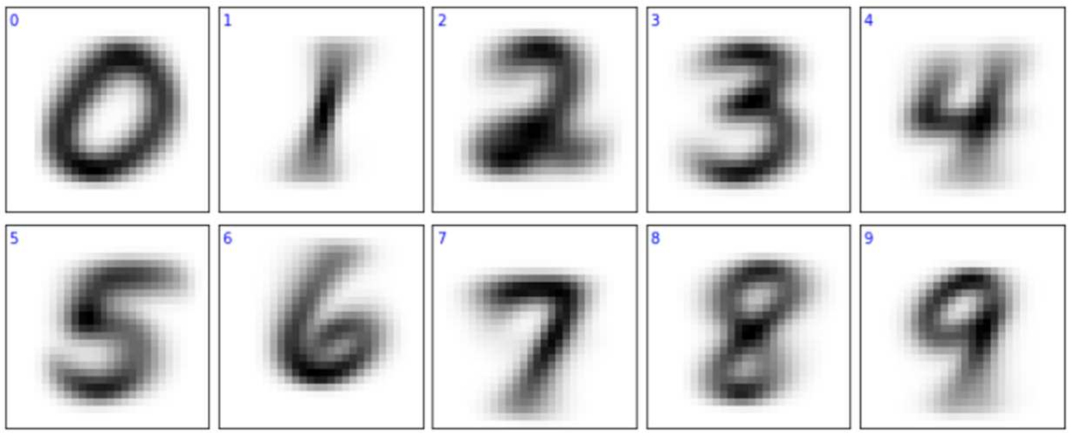
- Improvements?
  - **Idea 1**: classifying based on distance to the **center** of each class.
    - **What is the center of each class?**



  - **Idea 2**: using smaller samples of the dataset.

- Data Source: https://www.kaggle.com/c/digit-recognizer/data

center of each class



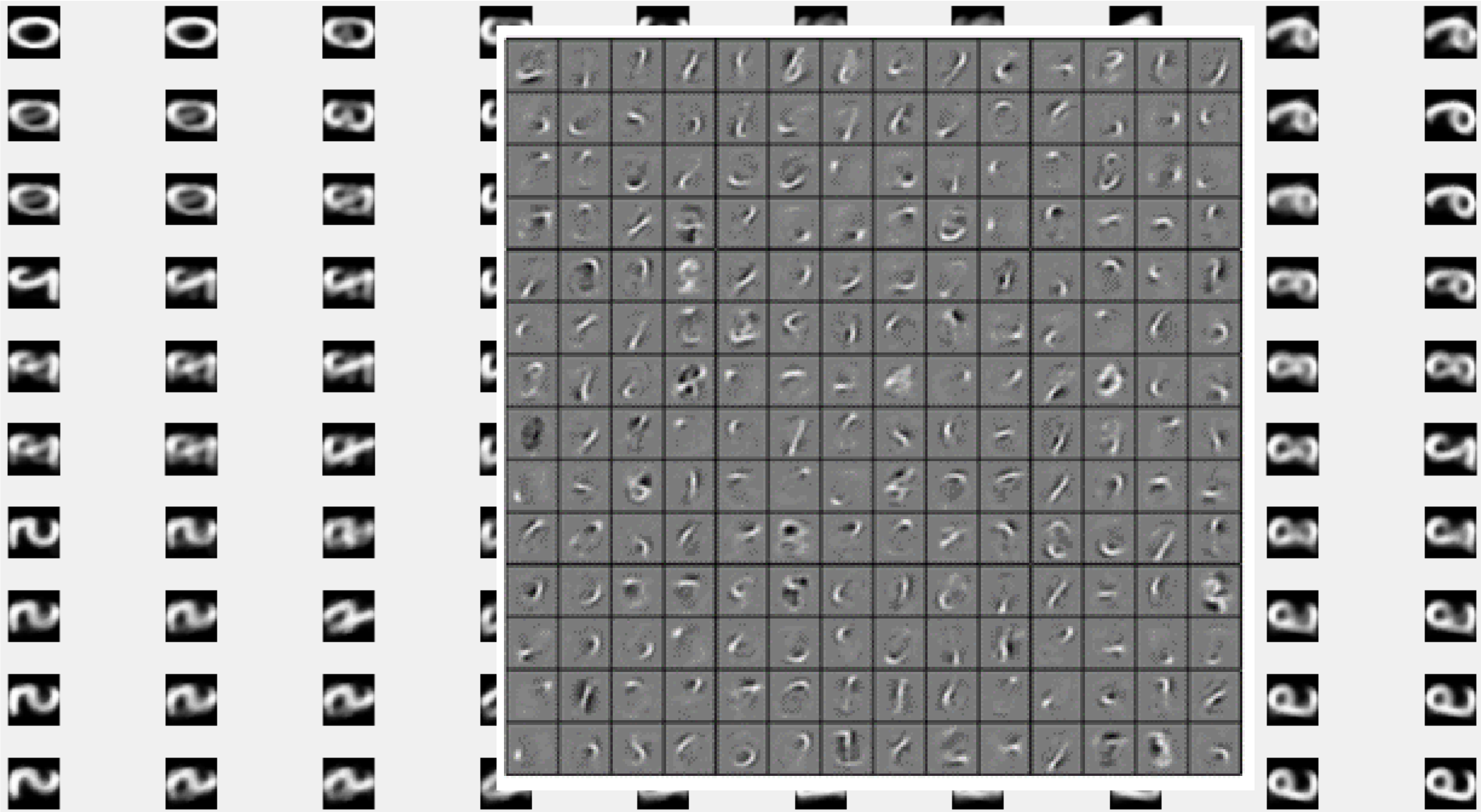
Digit Recognition, DM-02-18S

Alreshidi Abdulaziz, Yogita Singh
Bader Albulayhis, Sidi Mohamed,

Self Organizing Map (SOM)

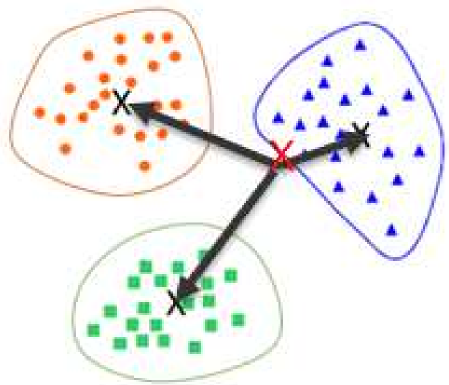
- MUCH “better” alternative➔ **Convolutional Neural Network** (CNN).
  - Less recognizable patterns. They are no longer centers. They are ***features***.

10×10 SOM neurons

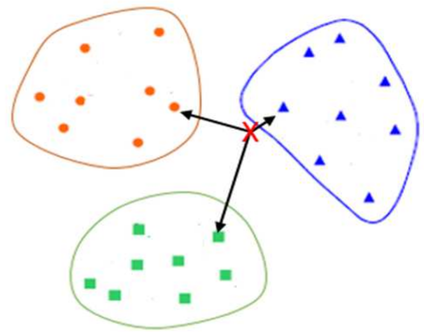


Idea 2: Using Smaller Samples from Training Data

- Much faster with accuracy tradeoff.

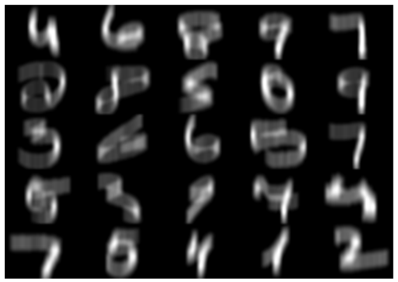


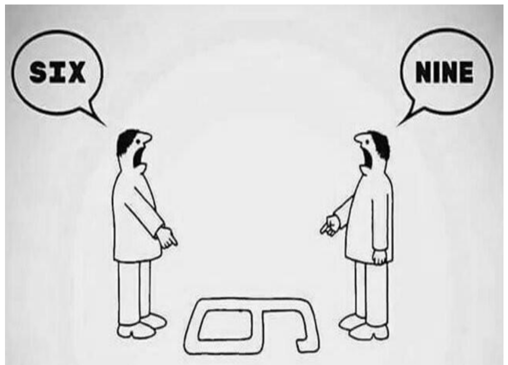
data size used in kNN	Accuracy	Exe Time
400 (1% of the data)	84%	1sec
2000 (5% of the data)	91%	20 sec
5000 (12.5% of the data)	93%	50 sec
10000 (25% of the data)	95%	2min



Digit Recognition, DM-02-18S

Alreshidi Abdulaziz, Yogita Singh
Bader Albulayhis, Sidi Mohamed,





Compare kNN Classification Quality to Other Methods

- Computing time is bit longer. But, kNN produce not bad result.
- How about executing time and quality of SVM or SVM+RBF?
- **How about advanced NN (i.e. CNN)???**

Algorithm	Accuracy (%)
Decision Tree	85
Naïve Bayes	82.55
KNN	96.0
Random Forest	96
MLP ← Regular NN	94

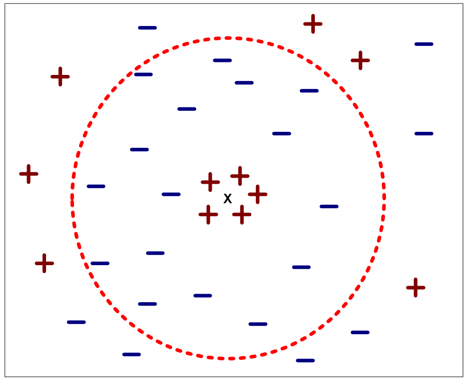
Digit Recognition, DM-02-18S

Alreshidi Abdulaziz, Yogita Singh
Bader Albulayhis, Sidi Mohamed,

- Some writings are difficult to classify…

2 1 4 4

Issues in ***k***NN, or Instance-Based Learning



- Difficult choosing right ***k*** value
  - If ***k*** is too small, sensitive to noise points
  - If ***k*** is too large, neighborhood may include points from other classes
- Numeric attributes with different scales.
  - Distance measures may be dominated by one of the attributes
    - Heights of persons vary from 1.5m to 1.8m. $$ of persons vary from $100K to $100B.
- Binary attributes.

R1 = (0 0 0)
R2 = (1 1 0)
R3 = (1 0 1)

- Categorical attributes. (e.g. diseases, states…)
  - Convert them to dummy variables…
  - **That's it??!!**
  - Before RBF, we **never** compare distance btwn records.
    - We only derive θ to compute $\theta^{T}X$.

1 0 0 0 0 0 0 0 0 0 0 0

0 1 0 0 0 0 0 0 0 0 0 0

0 0 0 0 0 0 0 0 0 0 0 1

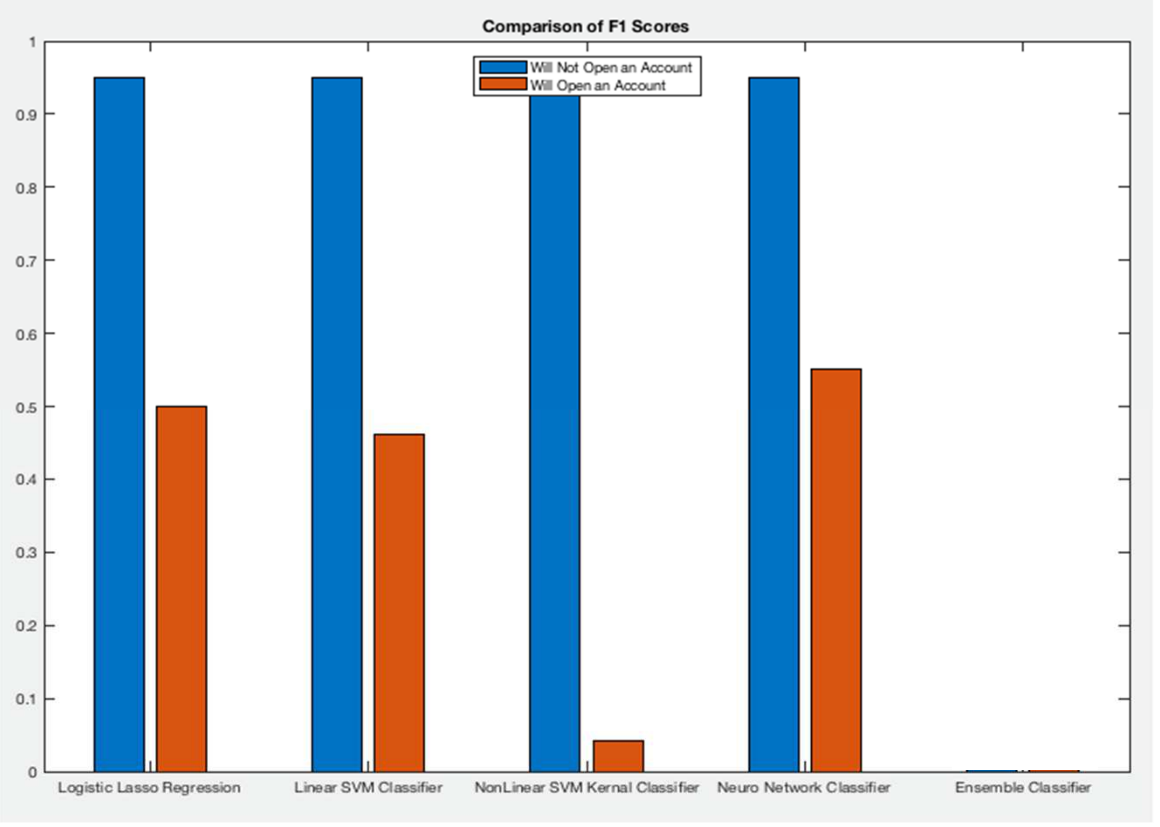
Lazy vs. Eager Learning

- Lazy evaluation
  - kNN or Naïve Bayes (**instance-based learner**).
  - **Less time training but more time predicting** **need to carry all instances**
  - Generalize **beyond** the **current** training data.

- Eager evaluation
  - Decision-tree, logistic, LDA, SVM, **SVM RBF** (**instance-based learner**)
  - **More time in training but less time in predicting**
  - **Commit** to a fixed / static model.

Reminder

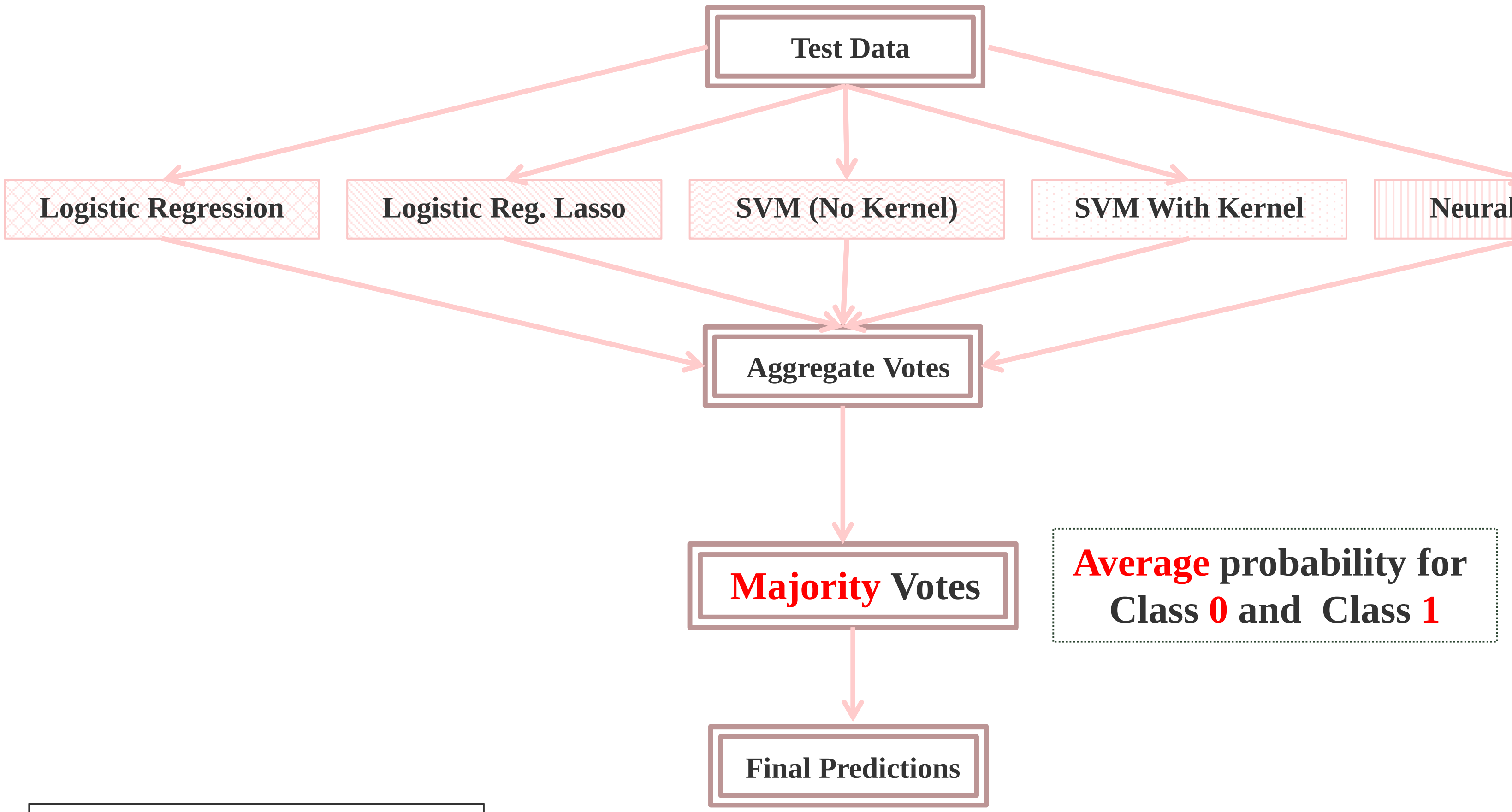
- We discussed many machine learning techniques.
- Try to use all or multiple methods.
- Not only you can compare their performance, but also… WHAT???



MDL-01 2018S, Bank Marketing

Terrence White
Ronald E Twite
Leela Sowjanya Chippada
Ahmad K Lubnani
Mowlid Abdillahi
Nathan Adams

Ensemble Hard / Soft Voting



MDL-01 2018S, Bank Marketing

Terrence White
Ronald E Twite
Leela Sowjanya Chippada
Ahmad K Lubnani
Mowlid Abdillahi
Nathan Adams

Characteristics of Classification Algorithms

- SVM
  - Speed & memory usage are good w/ few support vectors, poor if too many. Difficult to interpret how SVM classifies data w/ kernels. Easy for linear SVM.
- Naive Bayes
  - Speed & memory usage are good for simple distributions, but poor for kernel distributions and large data sets.
- Nearest Neighbor
  - Good predictions in low D, but poor predictions in high D. Need kd-trees for speed. Vars can be either continuous or categorical, not both.
- Discriminant Analysis
  - Accurate when **normal dist**. Otherwise, accuracy varies.

Algorithm	Predictive Accuracy	Fitting Speed	Prediction Speed	Memory Usage	Easy to Interpret	Handles Categorical Predictors
Trees	Medium	Fast	Fast	Low	Yes	Yes
SVM	High	Medium	*	*	*	No
Naive Bayes	Medium	**	**	**	Yes	Yes
Nearest Neighbor	***	Fast***	Medium	High	No	Yes***
Discriminant Analysis	****	Fast	Fast	Low	Yes	No
Ensembles	See Suggestions for Choosing an Appropriate Ensemble Algorithm and General Characteristics of Ensemble Algorithms					

Appendix

Why RBF ∈ ∞-Space?

- $\boldsymbol{K}(\boldsymbol{a}, \boldsymbol{b}) = \exp(-\|\boldsymbol{a} - \boldsymbol{b}\|^2) / 2\sigma^2 = e^{\frac{-\|a-b\|^2}{2\sigma^2}}$.

- $\boldsymbol{K}(x, y) = \exp(-\|x - y\|^2) = \exp(-(x_1 - y_1)^2 - (x_2 - y_2)^2)$

$$= \exp(-x_1^2 + 2x_1y_1 - y_1^2 - x_2^2 + 2x_2y_2 - y_2^2)$$

$$= \exp(-\|x\|^2) \times \exp(-\|y\|^2) \times \mathbf{exp(2x^Ty)}$$

$$k(x, y) = \exp(-\|x\|^2)\exp(-\|y\|^2)\boxed{\sum_{n=0}^{\infty}\frac{(2x^Ty)^n}{n!}}$$

- **Tylor series for *e*.**
  - Raise *x* & *y* to ***n***-dimension, divide it by ***n***-factorial, and sum to infinity.

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \frac{x^4}{4!} + \cdots$$