Machine Learning
Capacity and Performance Analysis
and R

Stephen O'Connell

May 3, 2011

Introduction

Brief Introduction to Machine Learning and Data Mining

- What, Why and How

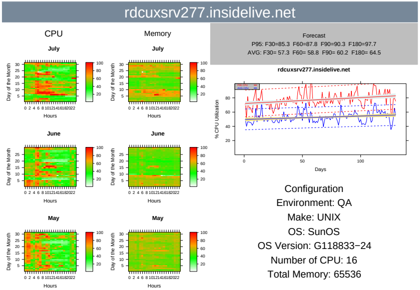
Introduction

Brief Introduction to Machine Learning and Data Mining

- What, Why and How

How can this be applied to Capacity and Performance Analysis

- Data driven
- Patterns



Introduction

Brief Introduction to Machine Learning and Data Mining

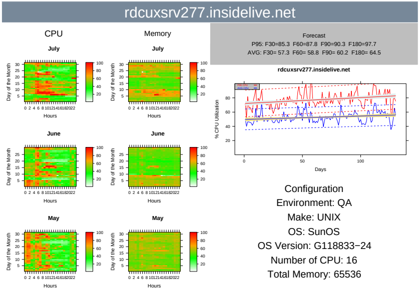
- What, Why and How

How can this be applied to Capacity and Performance Analysis

- Data driven
- Patterns

Example: Utilization Profiling in R

- Data Transformation
- Model Construction and Test
- Model Deployment



Machine Learning: Definition

There are Many: Here are a couple

Machine Learning: Definition

There are Many: Here are a couple

Definition:

Tom M. Mitchell provided a widely quoted definition: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.[1]

Machine Learning: Definition

There are Many: Here are a couple

Definition:

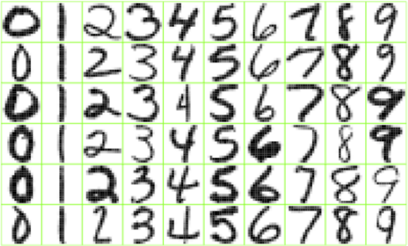
Tom M. Mitchell provided a widely quoted definition: A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.[1]

Definition:

The field of machine learning studies the design of computer programs able to induce patterns, regularities, or rules from past experiences. Learner (a computer program) processes data representing past experiences and tries to either develop an appropriate response to future data, or describe in some meaningful way the data seen. [2]

Example: Handwritten Digits

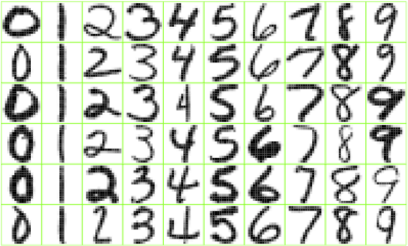
Elements of Machine Learning:



Example: Handwritten Digits

Elements of Machine Learning:

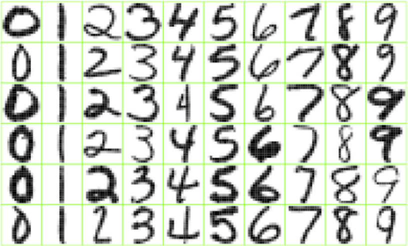
- Task T
  - recognizing and classifying handwritten words within images



Example: Handwritten Digits

Elements of Machine Learning:

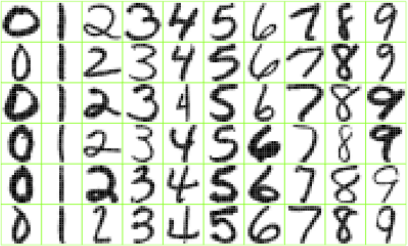
- Task T
  - recognizing and classifying handwritten words within images
- Performance P
  - percent of words correctly classified



Example: Handwritten Digits

Elements of Machine Learning:

- Task T
  - recognizing and classifying handwritten words within images
- Performance P
  - percent of words correctly classified
- Experience E
  - a database of handwritten words with given classifications



Methods

Supervised learning[4]:

Methods

Supervised learning[4]:

- Use a labeled (known) set of data to build models to perform classification or regression

Methods

Supervised learning[4]:

- Use a labeled (known) set of data to build models to perform classification or regression
- Use the model on new data to make predictions or describe the data

Methods

Supervised learning[4]:

- Use a labeled (known) set of data to build models to perform classification or regression
- Use the model on new data to make predictions or describe the data
- Supervised algorithms:
  - Linear Regression
  - Trees
  - Neural Networks
  - Support Vector Machines

Methods

Unsupervised learning[4]:

Methods

Unsupervised learning[4]:

- Find hidden structure in unlabeled (unknown) data

Methods

Unsupervised learning[4]:

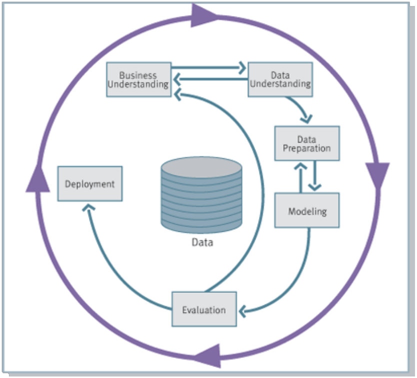
- Find hidden structure in unlabeled (unknown) data
- Unsupervised algorithms:
  - Kmeans
  - K Nearest Neighbor
  - Hierarchical Clustering
  - Association Rules
  - Principal Components

Machine Learning is a Process

Like application development

CRISP-DM, for example[3]:

- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation
- Deployment
- REPEAT AS NEEDED



Applications

Some notable applications:

Applications

Some notable applications:

- Spam Filtering – Yahoo

Applications

Some notable applications:

- Spam Filtering – Yahoo
- Fraud / Anomaly Detection - Credit Card

Applications

Some notable applications:

- Spam Filtering – Yahoo
- Fraud / Anomaly Detection - Credit Card
- Stock Predications / Trading Models

Applications

Some notable applications:

- Spam Filtering – Yahoo
- Fraud / Anomaly Detection - Credit Card
- Stock Predications / Trading Models
- Recommendation - Netflix

Applications

Some notable applications:

- Spam Filtering – Yahoo
- Fraud / Anomaly Detection - Credit Card
- Stock Predications / Trading Models
- Recommendation - Netflix
- Social Network Analysis – Facebook

Applications

Some notable applications:

- Spam Filtering – Yahoo
- Fraud / Anomaly Detection - Credit Card
- Stock Predications / Trading Models
- Recommendation - Netflix
- Social Network Analysis – Facebook
- Internet Search – Google

And you can make money!

Data Mining Competitions:

And you can make money!

Data Mining Competitions:

- Netflix – $1M

And you can make money!

Data Mining Competitions:

- Netflix – \$1M
- Heritage Health Prize - \$3M

And you can make money!

Data Mining Competitions:

- Netflix – \$1M
- Heritage Health Prize - \$3M
- Kaggle

Capacity Planning and Performance Analysis:

A simplified view of capacity planning:

Capacity Planning and Performance Analysis:

A simplified view of capacity planning:

- Hourly and Daily, Monday thru Friday stats

Capacity Planning and Performance Analysis:

A simplified view of capacity planning:

- Hourly and Daily, Monday thru Friday stats
- Peak and Average Daily Utilization

Capacity Planning and Performance Analysis:

A simplified view of capacity planning:

- Hourly and Daily, Monday thru Friday stats
- Peak and Average Daily Utilization
- Simple linear regression on peak and average utilization

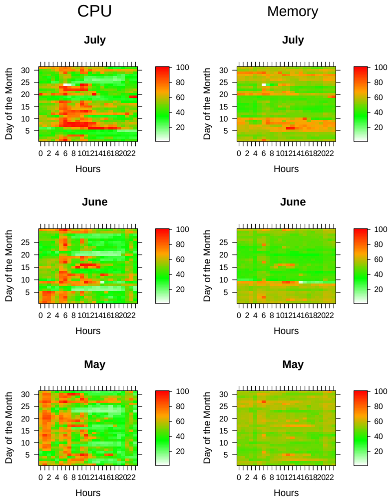
Capacity Planning and Performance Analysis:

A simplified view of capacity planning:

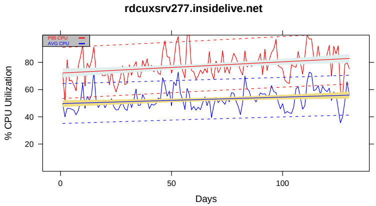
- Hourly and Daily, Monday thru Friday stats
- Peak and Average Daily Utilization
- Simple linear regression on peak and average utilization
- Extrapolate 30-60-90-180 days into the future.

Capacity Planning – Simplified View:

rdcuxsrv277.insidelive.net



Forecast
P95: F30=85.3 F60=87.8 F90=90.3 F180=97.7
AVG: F30= 57.3 F60= 58.8 F90= 60.2 F180= 64.5



Configuration
Environment: QA
Make: UNIX
OS: SunOS
OS Version: G118833−24
Number of CPU: 16
Total Memory: 65536

* One page Server Utilization, Forecast, and Configuration developed using R

Capacity Planning – Simplified View:

Put all forecasts into a spreadsheet and sort by 30, 60, 90, or 180 forecast to find top Utilized servers

	A	O	P	Q	R	S	T	U	V	W	X
1	*Host Capacity Forecast Report*		<0%	>80%	>90%						
2	server_name	days	avg_30_days	avg_60_days	avg_90_da	avg_180_c	p95_30_da	p95_60_da	p95_90_da	p95_180_c	avgm
3	rdclxsrv307.insidelive.net	129	145.44	169.73	194.02	266.89	140.25	161.71	183.18	247.56	5
4	calntmgt201.insidelive.net	129	131.69	146.18	160.68	204.16	132.62	147.04	161.46	204.73	3
5	rdclxsrv349.insidelive.net	126	117.36	132.64	147.91	193.73	117.97	133.18	148.4	194.05	5
6	sfouxsrv026.insidelive.net	129	112.19	135.61	159.03	229.29	116.24	139.45	162.66	232.3	3
7	sfouxsrv133.insidelive.net	129	98.66	116.56	134.47	188.17	103.55	121.31	139.06	192.33	6
8	yokuxsrv006.insidelive.net	129	93.27	110.22	127.17	178.01	107.51	122.72	137.94	183.57	8
9	calntscr001.insidelive.net	129	93.1	110.49	127.87	180.02	103.05	122.55	142.05	200.54	3
10	rdcuxsrv143.insidelive.net	129	89.35	92.05	94.75	102.85	95.98	97.89	99.79	105.52	3
11	rdcuxsrv161.insidelive.net	129	87.83	100.52	113.21	151.27	105.92	117.33	128.75	162.99	2
12	calntapp623.insidelive.net	42	75.69	106.86	138.02	231.53	85.12	118.05	150.97	249.75	5
13	rdcuxsrv017.insidelive.net	129	75.61	76.75	77.89	81.31	78.41	79.53	80.66	84.05	5
14	rmcuxsrv048.insidelive.net	129	75.29	75.96	76.62	78.62	100.03	100.06	100.09	100.19	1
15	rmcuxsrv099.insidelive.net	129	75.27	90.63	105.99	152.07	89.8	104.27	118.74	162.15	2
16	rdcuxsrv079.insidelive.net	129	74.61	85.59	96.57	129.52	102.6	119.14	135.68	185.31	4
17	rdcuxsrv134.insidelive.net	129	73.91	83.64	93.37	122.54	97.05	107.96	118.88	151.62	7
18	sfolxsrv085.insidelive.net	129	73.87	92.05	110.24	164.79	88.42	110.11	131.79	196.83	
19	sfolxsrv065.insidelive.net	115	73.36	83.59	93.82	124.51	73.87	83.81	93.74	123.56	
20	toklxsrv015.insidelive.net	129	73.24	82.98	92.72	121.94	74.12	83.93	93.75	123.2	4
21	rdcuxsrv276.insidelive.net	129	72.43	74.81	77.19	84.34	94.57	97.9	101.24	111.25	6
22	rdclxsrv267.insidelive.net	125	71.97	84.55	97.13	134.87	90.55	102.53	114.5	150.42	2
23	calntapp201.insidelive.net	129	71.85	78.5	85.15	105.11	88.94	95.47	101.99	121.57	7
24	calntesm412.insidelive.net	85	71.08	90.73	110.38	169.34	76.11	97.01	117.92	180.64	5
25	calntesm220.insidelive.net	26	70.72	105.89	141.06	246.58	87.74	127.98	168.21	288.92	6

Capacity Planning – Simplified View:

- Works well for stable business applications and environments.

Capacity Planning – Simplified View:

- Works well for stable business applications and environments.
- With 30-40-50 servers capacity planning for critical servers is straight forward.

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?
  - Many different environments, Production, Test, BCP, QA

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?
  - Many different environments, Production, Test, BCP, QA
  - Many different applications: database, web server, Quant, Hadoop, etc.

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?
  - Many different environments, Production, Test, BCP, QA
  - Many different applications: database, web server, Quant, Hadoop, etc.
  - Many different hardware platforms: Unix, Linux, Windows, VMWare, etc.

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?
  - Many different environments, Production, Test, BCP, QA
  - Many different applications: database, web server, Quant, Hadoop, etc.
  - Many different hardware platforms: Unix, Linux, Windows, VMWare, etc.
- Things are not stable and well formed – different applications have different utilization profiles, for different reasons.

Capacity Planning – Real world

- With 3k-4k-5k-+10k servers capacity planning is very difficult.
- Why?
  - Many different environments, Production, Test, BCP, QA
  - Many different applications: database, web server, Quant, Hadoop, etc.
  - Many different hardware platforms: Unix, Linux, Windows, VMWare, etc.
- Things are not stable and well formed – different applications have different utilization profiles, for different reasons.
- Exceptions occur

How can Machine Learning help?

- Lots and lots of data
  - System have many different components and each component has its own function and collection of metrics that determine performance
  - No problem is in isolation, there are many different sets of data that need to be correlated

How can Machine Learning help?

- Lots and lots of data
  - System have many different components and each component has its own function and collection of metrics that determine performance
  - No problem is in isolation, there are many different sets of data that need to be correlated
- Many different relationships server, storage, database, application...

How can Machine Learning help?

- Lots and lots of data
  - System have many different components and each component has its own function and collection of metrics that determine performance
  - No problem is in isolation, there are many different sets of data that need to be correlated
- Many different relationships server, storage, database, application...
- Lots of historical data (Capacity Database?)

How can Machine Learning help?

- Lots and lots of data
  - System have many different components and each component has its own function and collection of metrics that determine performance
  - No problem is in isolation, there are many different sets of data that need to be correlated
- Many different relationships server, storage, database, application...
- Lots of historical data (Capacity Database?)
- Many repeating and familiar patterns

How can Machine Learning help?

Elements of Machine Learning:

How can Machine Learning help?

Elements of Machine Learning:

- Task T
  - Classify resource utilization/performance

How can Machine Learning help?

Elements of Machine Learning:

- Task T
  - Classify resource utilization/performance
- Performance P
  - Filtered list of key utilization classes
  - Enhanced Monitoring
  - Smart Alerting

How can Machine Learning help?

Elements of Machine Learning:

- Task T
  - Classify resource utilization/performance
- Performance P
  - Filtered list of key utilization classes
  - Enhanced Monitoring
  - Smart Alerting
- Experience E
  - Historical data from a capacity database

Utilization Patterns

- Resource Utilization have patterns

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.
  - Cyclic – Highly variable workload, maybe some consistency like month-end or quant load.

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.
  - Cyclic – Highly variable workload, maybe some consistency like month-end or quant load.
  - Trend Increasing – Organic growth in the utilization.

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.
  - Cyclic – Highly variable workload, maybe some consistency like month-end or quant load.
  - Trend Increasing – Organic growth in the utilization.
  - Trend Decreasing – Reduced workload, application retirement.

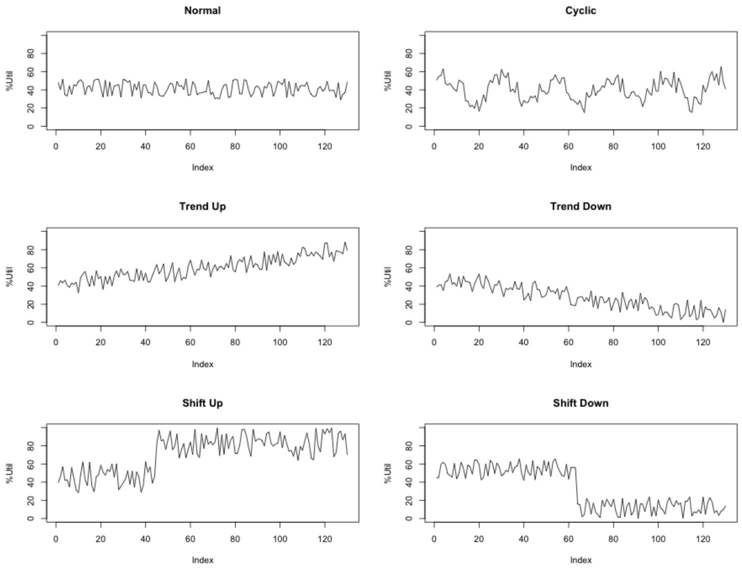
Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.
  - Cyclic – Highly variable workload, maybe some consistency like month-end or quant load.
  - Trend Increasing – Organic growth in the utilization.
  - Trend Decreasing – Reduced workload, application retirement.
  - Shift Upward – Sharp increase in processing, could be related to broken processes, cluster failover to stand-by server, and/or new application deployment.

Utilization Patterns

- Resource Utilization have patterns
- They are visual clues as to performance and future utilization
- These patterns can be grouped:
  - Normal – A flat utilization, stable environment, could be either consistently high/middle/low in its utilization.
  - Cyclic – Highly variable workload, maybe some consistency like month-end or quant load.
  - Trend Increasing – Organic growth in the utilization.
  - Trend Decreasing – Reduced workload, application retirement.
  - Shift Upward – Sharp increase in processing, could be related to broken processes, cluster failover to stand-by server, and/or new application deployment.
  - Shift Downward – Sharp decrease in processing, could be related to fixing processes, fail-back, or application retirement.

Utilization Patterns



What is R?

- Open source statistical programming language

What is R?

- Open source statistical programming language
- Great visualization packages

What is R?

- Open source statistical programming language
- Great visualization packages
- Many different modeling packages

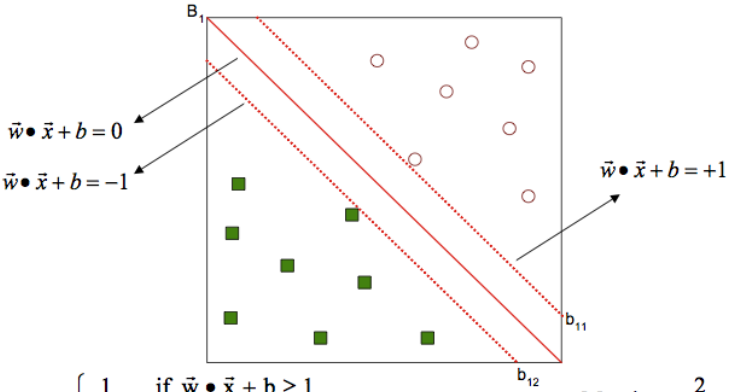
What is R?

- Open source statistical programming language
- Great visualization packages
- Many different modeling packages
- Many different machine learning packages

What is R?

- Open source statistical programming language
- Great visualization packages
- Many different modeling packages
- Many different machine learning packages
- Almost a complete solution for building machine learning tools – scaling is an issue, i.e. the problem has to fit in memory.

Support Vector Machine



$$f(\vec{x}) = \begin{cases} 1 & \text{if } \vec{w} \bullet \vec{x} + b \geq 1 \\ -1 & \text{if } \vec{w} \bullet \vec{x} + b \leq -1 \end{cases}$$

$$\text{Margin} = \frac{2}{\|\vec{w}\|^2}$$

Data Considerations

- Performance and Utilization data is time series

```
DateTime      Server       AverageCPU
2011-05-01    webserver    95
2011-05-02    webserver    90
2011-05-03    webserver    85
2011-05-04    webserver    95
2011-05-05    webserver    94
```

- ML data format is a matrix with the general form Y, X1, X2,...Xn
- Need to convert time series to matrix

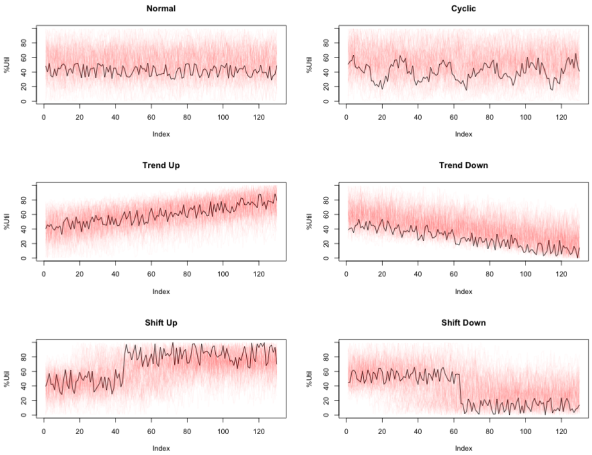
```
Y            X1    X2    X3    X4    X5
webserver    95    90    85    95    94
```

- Data needs to be consistent and well formed, no missing or bad data

SVM Demonstration

- Data is generated, `prototypes.R`
- `helperFunctions.R`
  - `createData`
  - `confusionM`
  - `printMissClassified`
- `demo_1.R` Builds an initial SVM model, tunes the model and classifies new data with the model
- `demo_2.R` Improves the accuracy of the initial model

Generated Data



- Datasets contain 100 of each type of pattern, i.e. 600 servers
- There are 130 X data points/features representing 180 days, Monday thru Friday
- Randomly generated...

First Predictive Model

Create the first model

```
###---------------------------------- OUT OF THE BOX ----------------------------------
## GET DATA
Ynew <- dget("Y_7")
data <- createData(Ynew)

## SPLIT TO x and Y
x <- subset(data, select =-class)
y <- data$class

## BUILD MODEL
model <- svm(class ~ ., data = data)
summary(model)

Call:
svm(formula = class ~ ., data = data)

Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.007692308

Number of Support Vectors:  545
 ( 95 100 89 80 88 93 )

Number of Classes:  6

Levels:
 Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
```

First Predictive Model

Check the models accuracy

```
> ## PREDICTIONS
pred <- predict(model, x)

# CHECK ACCURACY:
confusionM(pred, y)

Predicted Values:
Yp
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
       94       109         90         92        108        107
Y values:
Y
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
      100       100        100        100        100        100
Confusion Matrix:
            Yp
Y            Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic         90     10         0       0         0       0
  Normal          2     98         0       0         0       0
  ShiftDown       2      0        90       0         8       0
  ShiftUp         0      1         0      91         0       8
  TrendDown       0      0         0       0       100       0
  TrendUp         0      0         0       1         0      99
Accuracy =  0.9466667[1] 0.9466667
```

First Predictive Model

Model Tuning:

```
> obj <- tune.svm(class~., data = data, gamma = 2^(-1:1), cost = 2^(2:4))
> summary(obj)

Parameter tuning of svm:

- sampling method: 10-fold cross validation

- best parameters:
 gamma cost
   0.5    4

- best performance: 0.8883333

- Detailed performance results:
  gamma cost     error dispersion
1   0.5    4 0.8883333 0.03604695
2   1.0    4 0.9000000 0.03142697
3   2.0    4 0.9000000 0.03142697
4   0.5    8 0.8883333 0.03604695
5   1.0    8 0.9000000 0.03142697
6   2.0    8 0.9000000 0.03142697
7   0.5   16 0.8883333 0.03604695
8   1.0   16 0.9000000 0.03142697
9   2.0   16 0.9000000 0.03142697
```

First Predictive Model

Re-run the training data with tuned parameters

```
> ###-------------------------------- AFTER TUNING --------------------------------
## NEW MODEL WITH COST AND GAMMA
model <- svm(class ~ ., data = data, cost=2.25, gamma=.01)

## RE-DO THE PREDICTION
pred <- predict(model, x)

# CHECK ACCURACY
confusionM(pred, y)

 Predicted Values:
Yp
    Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
        98       102         98         98        102        102
Y values:
Y
    Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
       100       100        100        100        100        100
 Confusion Matrix:
           Yp
Y            Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic         98      2         0       0         0       0
  Normal          0    100         0       0         0       0
  ShiftDown       0      0        98       0         2       0
  ShiftUp         0      0         0      98         0       2
  TrendDown       0      0         0       0       100       0
  TrendUp         0      0         0       0         0     100
 Accuracy =  0.99[1] 0.99
```

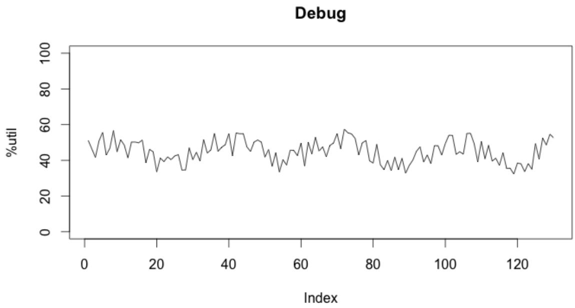
First Predictive Model

Missclassification Analysis

```
> printMissClassified(pred, y)

   Predicted     Actual  observation
1     Normal     Cyclic          156
2     Normal     Cyclic          177
3    TrendUp    ShiftUp          423
4    TrendUp    ShiftUp          452
5  TrendDown  ShiftDown          553
6  TrendDown  ShiftDown          586

> plot(t(data[156,2:131]), type='l',main="Debug",ylab="%util", ylim=c(0,100))
```



First Predictive Model

Use the model to classify new data

```
###———————————————————— NEW DATA ————————————————————
## READ IN DATA THAT MODEL HAS NOT SEEN
Ynew <- dget("Y_6")
data <- createData(Ynew)
## SPLIT TO X and Y
x <- subset(data, select = -class)
y <- data$class
## PREDICT CLASS USING PREVIOUSLY CREATED MODEL
pred <- predict(model, x)

# CHECK ACCURACY
confusionM(pred, y)
Predicted Values:
Yp
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
       96       107         99         86        102        110
Y values:
Y
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
      100       100        100        100        100        100
Confusion Matrix:
           Yp
Y           Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic        69     23         6       0         2       0
  Normal        20     80         0       0         0       0
  ShiftDown      5      0        85       0        10       0
  ShiftUp        0      4         0      77         0      19
  TrendDown      2      0         8       0        90       0
  TrendUp        0      0         0       9         0      91
Accuracy =  0.82[1] 0.82
```

How Can we improve on these results?

- More tuning of the cost and gamma parameters?

How Can we improve on these results?

- More tuning of the cost and gamma parameters?
- Is the data representative of the real world?

How Can we improve on these results?

- More tuning of the cost and gamma parameters?
- Is the data representative of the real world?
- More training data in the model?

How Can we improve on these results?

- More tuning of the cost and gamma parameters?
- Is the data representative of the real world?
- More training data in the model?
- Less training data in the model?

How Can we improve on these results?

- More tuning of the cost and gamma parameters?
- Is the data representative of the real world?
- More training data in the model?
- Less training data in the model?
- Model overfitting?

How Can we improve on these results?

- More tuning of the cost and gamma parameters?
- Is the data representative of the real world?
- More training data in the model?
- Less training data in the model?
- Model overfitting?
- Are we using the right algorithm, the right way?

Improved Predictive Model

Add more data to the Model Build

```
> ###———————————————————— OUT OF THE BOX ————————————————————
## ADD MORE DATA TO THE MODEL BUILD
Ynew <- dget("Y_7")
d <- createData(Ynew)
data <- d
Ynew <- dget("Y_5")
d <- createData(Ynew)
data <- rbind(data, d)
Ynew <- dget("Y_4")
d <- createData(Ynew)
data <- rbind(data, d)
Ynew <- dget("Y_3")
d <- createData(Ynew)
data <- rbind(data, d)
Ynew <- dget("Y_2")
d <- createData(Ynew)
data <- rbind(data, d)
Ynew <- dget("Y_1")
d <- createData(Ynew)
data <- rbind(data, d)

## SPLIT TO x and Y
x <- subset(data, select = -class)
y <- data$class
```

Improved Predictive Model

Add more data to the Model Build, cont.

```
> ## BUILD MODEL
model <- svm(class ~ ., data = data)
summary(model)


Call:
svm(formula = class ~ ., data = data)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.007692308

Number of Support Vectors:  2475

 ( 479 550 361 346 374 365 )


Number of Classes:  6

Levels:
 Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
```

Improved Predictive Model

Add more data to the Model Build, cont.

```
> ## PREDICTIONS
pred <- predict(model, x)

# CHECK ACCURACY:
confusionM(pred, y)

 Predicted Values:
Yp
   Cyclic    Normal ShiftDown   ShiftUp TrendDown   TrendUp
      576       627       580       582       619       616
Y values:
Y
   Cyclic    Normal ShiftDown   ShiftUp TrendDown   TrendUp
      600       600       600       600       600       600
 Confusion Matrix:
           Yp
Y            Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic        529     61         4       0         6       0
  Normal         37    563         0       0         0       0
  ShiftDown       8      1       566       0        25       0
  ShiftUp         0      2         0     576         0      22
  TrendDown       2      0        10       0       588       0
  TrendUp         0      0         0       6         0     594
 Accuracy =  0.9488889[1] 0.9488889
```

Improved Predictive Model

Tune the new model

```
> ###--------------------------------------------  AFTER TUNING ---------------------------------------------
## NEW MODEL WITH COST AND GAMMA
model <- svm(class ~ ., data = data, cost=2.25, gamma=.01)

## RE-DO THE PREDICTION
pred <- predict(model, x)

# CHECK ACCURACY
confusionM(pred, y)
Predicted Values:
Yp
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
      568       626        596        594        610        606
Y values:
Y
   Cyclic    Normal  ShiftDown    ShiftUp  TrendDown    TrendUp
      600       600        600        600        600        600
Confusion Matrix:
           Yp
Y            Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic        566     28         3       0         3       0
  Normal          2    598         0       0         0       0
  ShiftDown       0      0       593       0         7       0
  ShiftUp         0      0         0     593         0       7
  TrendDown       0      0         0       0       600       0
  TrendUp         0      0         0       1         0     599
Accuracy =  0.9858333[1] 0.9858333
```

Improved Predictive Model

Use the new model to classify new data

```
> ###---------------------------------------- NEW DATA ----------------------------------------
## READ IN DATA THAT MODEL HAS NOT SEEN
Ynew <- dget("Y_6")
data <- createData(Ynew)
## SPLIT TO X and Y
x <- subset(data, select = -class)
y <- data$class
## PREDICT CLASS USING PREVIOUSLY CREATED MODEL
pred <- predict(model, x)

# CHECK ACCURACY
confusionM(pred, y)
Predicted Values:
Yp
   Cyclic    Normal ShiftDown   ShiftUp TrendDown   TrendUp
       84       114        95        93       109       105
Y values:
Y
   Cyclic    Normal ShiftDown   ShiftUp TrendDown   TrendUp
      100       100       100       100       100       100
Confusion Matrix:
           Yp
Y           Cyclic Normal ShiftDown ShiftUp TrendDown TrendUp
  Cyclic        72     22         3       0         3       0
  Normal        10     90         0       0         0       0
  ShiftDown      2      0        89       0         9       0
  ShiftUp        0      2         0      88         0      10
  TrendDown      0      0         3       0        97       0
  TrendUp        0      0         0       5         0      95
Accuracy =  0.885[1] 0.885
```

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.
- Experimentation is required; no one best solution. There are always trade-offs.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.
- Experimentation is required; no one best solution. There are always trade-offs.
- Continually monitor model performance: is the real world drifting?

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.
- Experimentation is required; no one best solution. There are always trade-offs.
- Continually monitor model performance: is the real world drifting?
- Need to have model measurement and validation processes.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.
- Experimentation is required; no one best solution. There are always trade-offs.
- Continually monitor model performance: is the real world drifting?
- Need to have model measurement and validation processes.
- Change control of a new model, what, why and how.

Model Deployment Considerations

- Need to create the training data from "real" data; lots of labeling required.
- Patterns and labeling need to be consistent with objectives.
- Training data and "New" data need to be well formed, and consistent.
- Need to consider rare observations: anomaly detection.
- Experimentation is required; no one best solution. There are always trade-offs.
- Continually monitor model performance: is the real world drifting?
- Need to have model measurement and validation processes.
- Change control of a new model, what, why and how.
- Models are guides, not the answer.

Thank You!

E-mail: sao@saoconnell.com
Phone: 925-330-4350

Example code and slides available: ??

References:

- Wikipedia
  http://en.wikipedia.org/wiki/Machine_learning#Definition
- Vucetic, Slobodan http://www.ist.temple.edu/~vucetic/cis526fall2003/lecture1.pdf
- CRoss Industry Standard Process for Data Mining
  http://www.crisp-dm.org/Process/index.htm
- Trevor Hastie, Robert Tibshirani, Jerome Friedman, " The Elements of Statistical Learning:
  Data Mining, Inference, and Prediction."