1. Machine learning
2. Data mining
3. Pattern Recognition

* Computational science
* Domain: Software, Technology, manufacturing, construction and real estate.
* Statistics

Problem

Build infrastructure to solve the problem.

Frame the problem.

Attributes – real-valued, categorical. (impacts math that can be used)

80% - scrub data, then..normalize the data.

Data reduction

Use principal component analysis to come up with a low rank,

Go to covaraiance, find the top k eigen values

|  |  |  |  |
| --- | --- | --- | --- |
| (related to class) | Attributes/features |  |  |
| Instances | Vector (a,b,c…) | | |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Analyze patterns, see how they correlate

Happy with number of instances, and attributes then analyze

**Build the model to explain the data (eg. mixed models: expectation)**

(not too complex, can’t generalize – high bais, high variance, etc.)

Data set – divide (training, cross validation, test)

Training (incl. truth data)

Look at similarities:

1. between vectors (dot product cosine between vectors)
2. a dot b/ sq rt (a-norm) \* sq rt (b-norm)-> closer to one yes.

b) in max (a,b), = sum min(a,b) /sum Max(a,b)

If a and b ar probilities distribution then have to pick a different similarity measure.

Logistic regression, Nueral networks – very flexible

N-fold cross validation

n-fold cross validation

Train n-1

Test on holdout

**Eg. Classification**

[Preprocess]

Attributes

-Select to visualize all

[Visualize]

Visualize to look at whether the attributes separate well.

Tab [Classify]

Classify (choose algorithm)

Select cross-validation 10-folds

Look at correctly classified instances [%]

**Eg. Spam /not spam**

P(c=spam|x=x1,x2….)

Model?

100,000 emails (covariance) i.e. 100k squared- too expensive

If conditionally dependent, I don’t care about covariance.

Naïve-bais (model each word with its own Gaussian distribution), independent assumption of attributes=Naïve

Go from counts to occur or not occur (0/1) no covariance matrix required.

(Weka, file: spambase)

Class = 2788-no spam, 1813-spam

**Neural Networks**

Simple network – perceptron

Step function

3 inputs (3 weights, w1,w2,w3) ) + intercept/bais termX0=1 (W0)->step function-> output

Takes the dot product of inputs and weights and if crosses a threshold it fires, it is 1, -1

Need to adjust the weights to get a better truth value

Use error, determine how much each weight contributed to error

Then do back-propagation to adjust the weights

Logistic function is differentiable, therefore us instead of step function

If you go too deep with layers, backpropagation leads to zeros.

7/8 hidden layers (sufficient for motion)

2 layers are generally

**Theano toolkit (Deep learning)**

**Weka eg. Neural Networks: Hepatities (Multi-layer Perceptron.)**

**(right click on “Choose” field) to adjust factors (learning rate)**