

22. Smart Meters

金力 Li Jin

li.jin@sjtu.edu.cn

上海交通大学密西根学院

Shanghai Jiao Tong University UM Joint Institute



上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

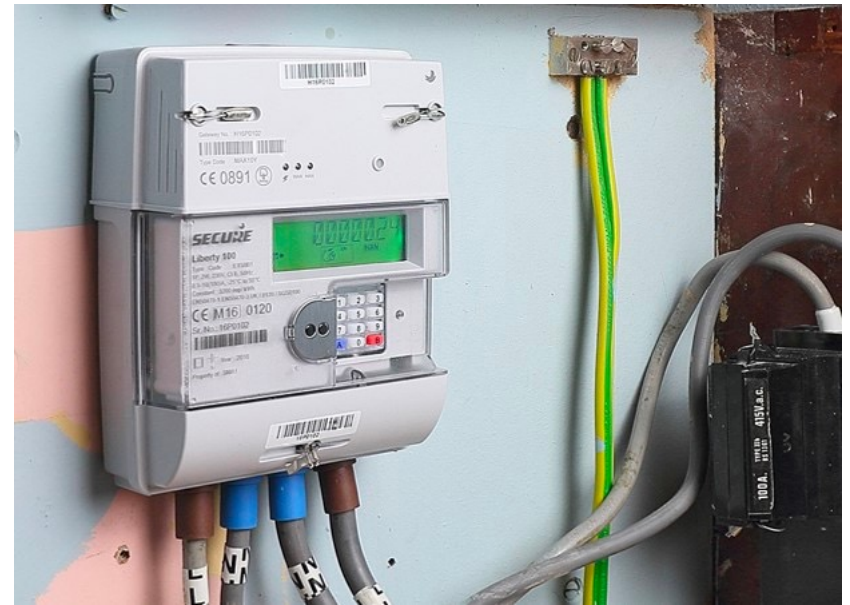
Outline

- Smart meters
- Linear classification
- Neural networks

Dong M, Meira P C M, Xu W, et al. An event window based load monitoring technique for smart meters[J]. IEEE transactions on smart grid, 2012, 3(2): 787-796.

Electricity load monitoring

- Objective: track use of home appliances
- Motivation: save energy and conduct demand response programs
- Input: load power curve
- Tool: linear classification
- Output: active appliances



Background

- The increased public awareness of energy conservation in recent years has created a huge interest in home energy consumption monitoring.
- Consumers show substantial interest in tools that can help them manage their household energy use and expenses.
- A critical link to address this need is the smart meters.
- However, the smart meters currently available in the market can only provide the energy consumption data of a whole house.



Challenge

- Smart meters cannot tell which appliances in the household consume the most energy or are least efficient.
- Also, to take full advantage of time-of-use rates, householders need to be informed of their usage patterns.
- Such information is essential for a household to make sound energy saving decisions and participate in utility demand response programs.



Tow solutions

- In response to this need, two research directions have emerged.
- One is to connect energy monitors to **individual appliance** of interest and to communicate the recorded data to a data concentrator.
- While such a sensor network-based system can provide accurate measurement of appliance energy consumption, it can be costly and complex to implement.
- The second direction is to identify and track major home appliances based on the **total signal** collected by utility meters, which is called nonintrusive load monitoring (NILM) method.
- Compared to the former, the NILM direction is more attractive to customers and utilities due to its high cost efficiency and less effort on installation.

Nonintrusive load monitoring

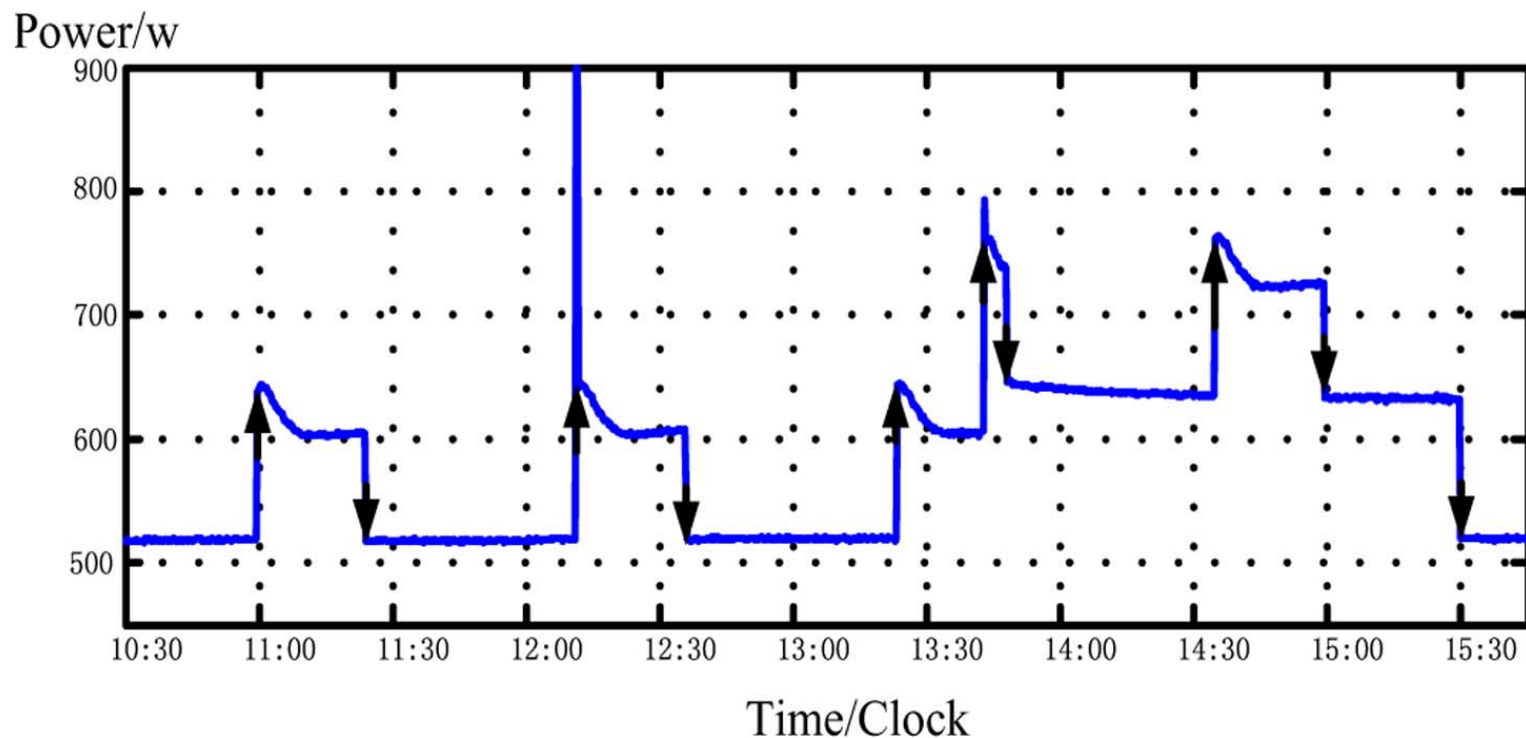
- The problem to be solved by NILM approach can be stated as follows:
- All the load signals aggregate at the entry point of a house as $P(t)$ and NILM algorithms do the reverse—decode the overall signal into various components $P_i(t)$ that are attributed to specific loads (appliances)
$$P(t) = P_1(t) + P_2(t) + \cdots + P_n(t)$$
- Only large appliances can (and should) be detected; phone chargers etc. are not interesting.
- What are the major power consumers?

Large appliances

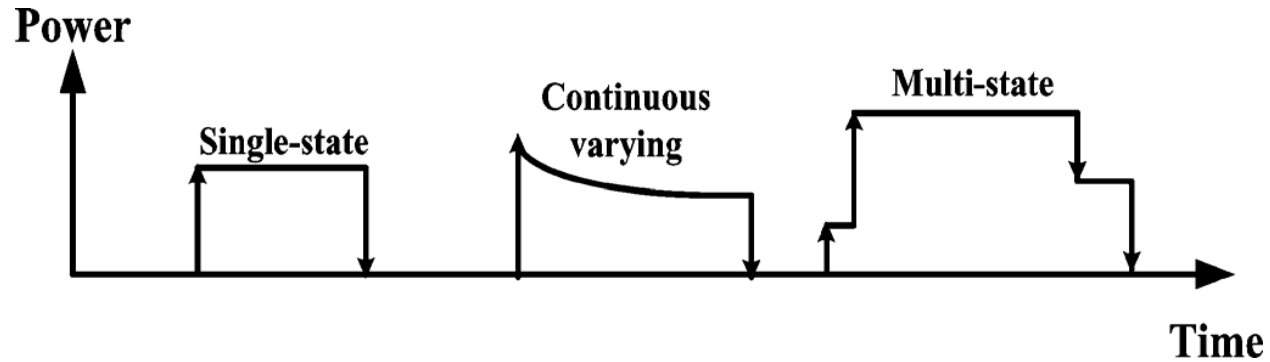


Data

- Appliance events: edges
- Gradual varying power demand between edges



Load types



Load type	Examples	Event	Power demand
Single-state	Light bulb; Toaster	ON=OFF	Flat
Continuous varying	Fridge; Freezer	ON≠OFF	Varying
Multi-state	Furnace; Washer	Multiple events	Varying or flat

Predictors and response

- Input: shape of power curve
 - edge signatures
 - sequence signatures
 - trend signatures
 - time/duration signatures
 - phase signatures
- Output: which appliance is active
 - We can have multiple appliances ON



Outline

- Smart meters
- Linear classification
- Neural networks

Classification problem

- Like most engineering problems, the most difficult part is setting up the model.
- That is also the most effort-consuming part.
- Specifically, generating predictors is the main challenge
 - On the one hand, they should capture what intuitively make sense
 - On the other hand, they are in formats compatible with classification algorithms
- A typical paradigm: human generate the predictors, while computer trains the model.

Linear classification

- Predictors: $x = (x_1, x_2, \dots, x_p)$
- k features of a sample
- Response: $G \in \{1, 2, \dots, K\}$
- Every sample belongs to one out of K classes
- Classification problem: given predictors $X = x$, what is the probability that this sample belongs to a particular class?

$$\Pr\{G = k | X = x\} = ?$$

- If we can compute this probability, then our prediction of class is

$$\hat{G} = \arg \max_k \Pr\{G = k | X = x\}$$

Linear regression of an indicator matrix

- Predictors: $x = (x_1, x_2, \dots, x_p)$
- Response: $y = (y_1, y_2, \dots, y_K)$
 - If an observation is in class k , then $y_k = 1$ and $y_j = 0$ for $j \neq k$
- Assume that the indicator variables y_1, y_2, \dots, y_K linearly depend on the predictors
 - $y_1 = \beta_{1,0} + \beta_{1,1}x_1 + \beta_{1,2}x_2 + \dots + \beta_{1,p}x_p$
 - $y_2 = \beta_{2,0} + \beta_{2,1}x_1 + \beta_{2,2}x_2 + \dots + \beta_{2,p}x_p$
 - ...
 - $y_K = \beta_{K,0} + \beta_{K,1}x_1 + \beta_{K,2}x_2 + \dots + \beta_{K,p}x_p$
- In matrix form
 - $y = X^T B$
 - y is $K \times 1$, B is $(p + 1) \times K$, X is $(p + 1) \times 1$

Linear regression of an indicator matrix

- How to determine coefficient matrix B ?
- Suppose that we have n observations

- RSS =

$$\sum_{i=1}^n \sum_{k=1}^K (y_{i,k} - \beta_{k,0} + \beta_{k,1}x_1 + \beta_{k,2}x_2 + \cdots + \beta_{k,p}x_p) \\ = (Y - X^T B)^T (Y - X^T B)$$

- Minimize RSS

$$\hat{B} = (X^T X)^{-1} X^T Y$$

- Fitted classes

$$\hat{Y} = X^T \hat{B} = X(X^T X)^{-1} X^T Y$$

Prediction

A new observation with input x is classified as follows

1. Compute the fitted output $\hat{f}(x) = (1, x^T)\hat{B}$, a K vector
2. Identify the largest component and classify accordingly

$$\hat{G}(x) = \arg \max_k \hat{f}_k(x)$$

Why?

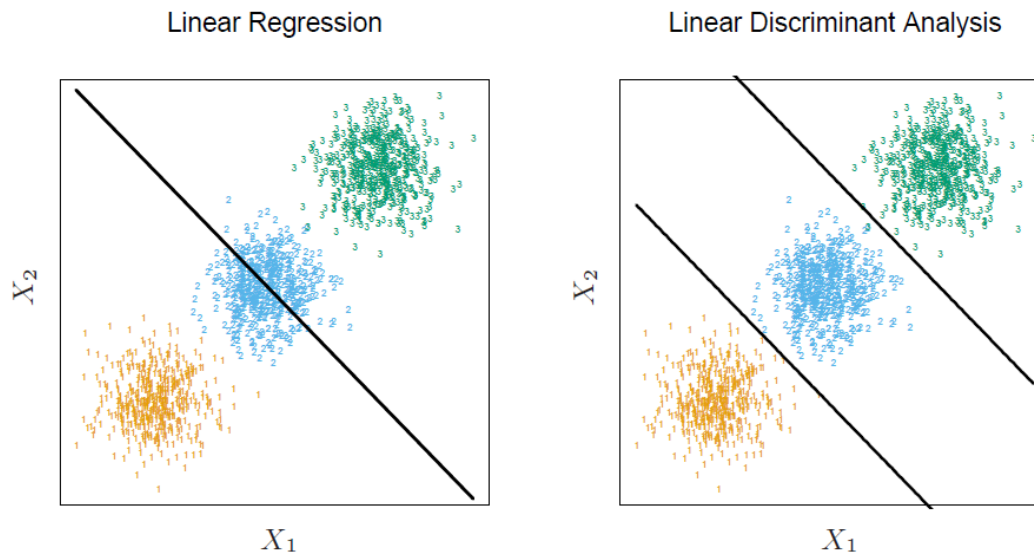
- A rather formal justification: y_k is a random variable such that

$$E[\hat{y}_k | X = x] = \Pr\{G = k | X = x\}$$

- That is, $\hat{y}_k \approx \Pr\{G = k | X = x\}$
- Illustration for two-class problem

LR not always work

- It seems that $\hat{y}_k \approx \Pr\{G = k|X = x\}$
- However, $\hat{y}_k = \beta_{k,0} + \beta_{k,1}x_1 + \beta_{k,2}x_2 + \cdots + \beta_{k,p}x_p$ can be < 0 or > 1 , which is not allowed for probabilities
- Another serious problem: **masking**
- Illustration for three-class problem



Linear discriminant analysis*

- Consider N observations
- Predictor vector x
- Class g
- Define
 - $\hat{\pi}_k = N_k/N$, where N_k is the number of class- k observations
 - $\hat{\mu}_k = \sum_{g_i=k} x_i / N_k$
 - $\hat{\Sigma} = \sum_{k=1}^K \sum_{i:g_i=k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T / (N - K)$

- Linear discriminant function

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$$

- Classification

$$G(x) = \arg \max_k \delta_k(x)$$

Logistic regression*

- Logit function

$$\Pr\{G = k | X = x\} = \frac{\exp(\beta_{k0} + \beta_k^T x)}{\sum_{l=1}^K \exp(\beta_{l0} + \beta_l^T x)}$$

- Classification

$$G(x) = \arg \max_k \frac{\exp(\beta_{k0} + \beta_k^T x)}{\sum_{l=1}^K \exp(\beta_{l0} + \beta_l^T x)}$$

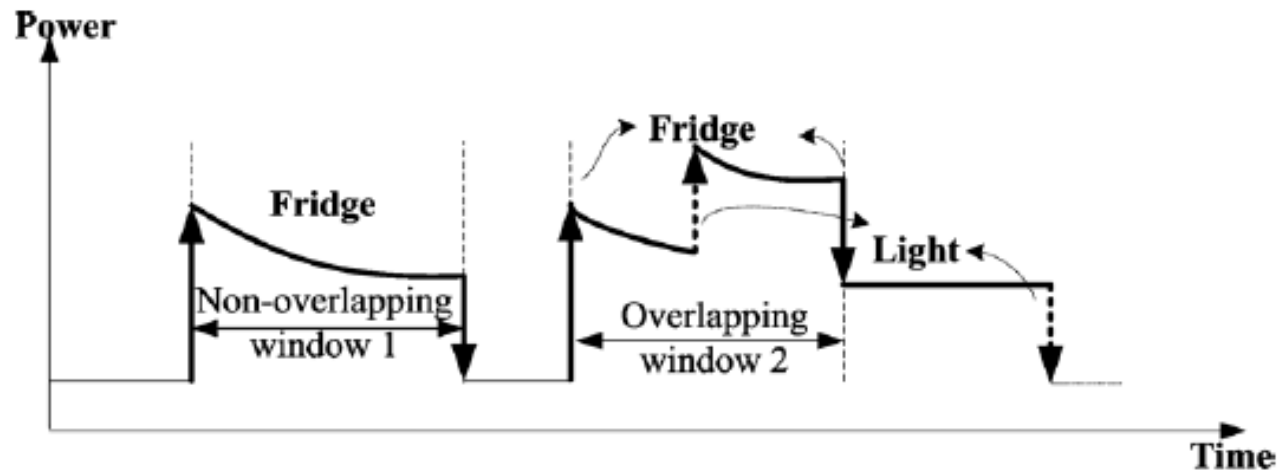
- Process of fitting coefficients β_{ki} : called logistic regression
- Use maximum likelihood

Summary of linear classification*

- Predictor x
- Find some function $f_k(x)$ such that
$$f_k(x) \approx \Pr\{G = k | X = x\}$$
- For a new observation with input x , classify as follows
$$\hat{G}(x) = \arg \max_k f_k(x)$$
- $f_k(x)$ is linear in x -> linear regression
- $f_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log \pi_k$ -> LDA
- $f_k(x) = \frac{\exp(\beta_{k0} + \beta_k^T x)}{\sum_{l=1}^K \exp(\beta_{l0} + \beta_l^T x)}$ -> logistic regression

Event windows

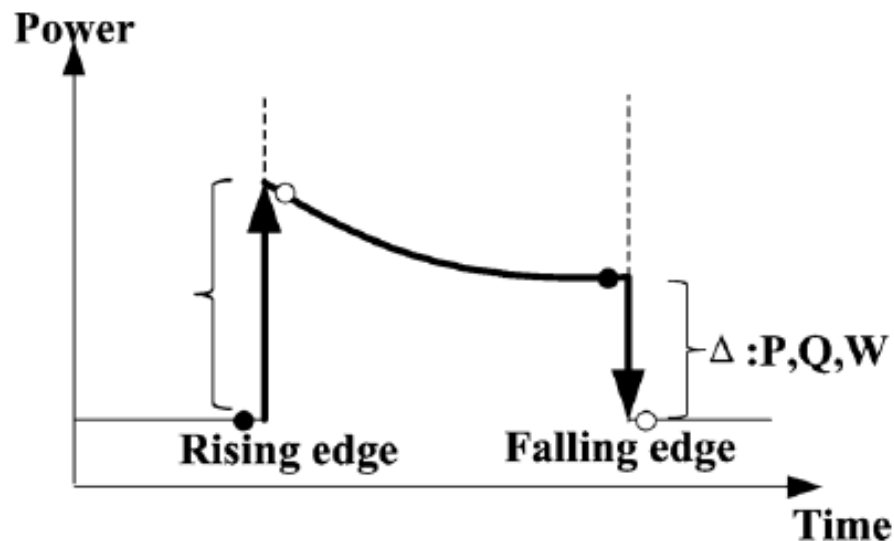
- An event window is defined as the collection of all signatures between any pair of rising/falling step-changes (events) of the power demand as measured by the smart meter.



- Overlapping: most appliances
- Non-overlapping: short-duration or always-on

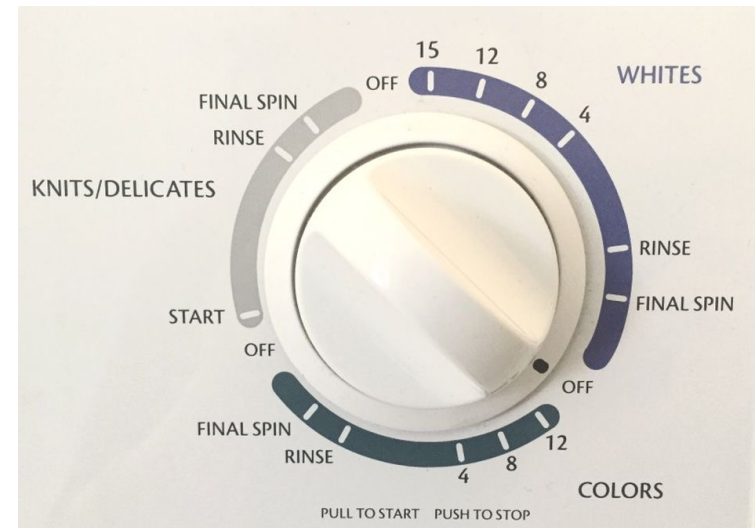
Edge signature

- An edge refers to the event of the operating state of an appliance, which be a step change in its power demand.
- The edge can be either rising or falling.
- Each edge can be characterized by the changes in power (P), reactive power (Q), and current waveform (W)

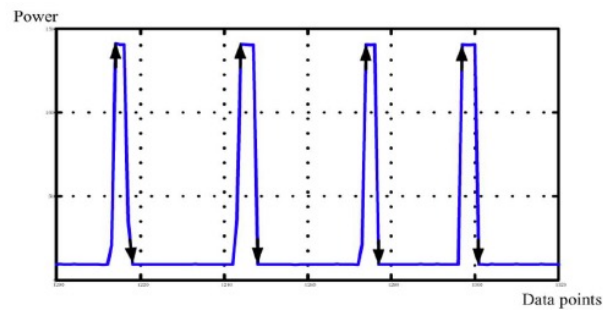


Sequence signature

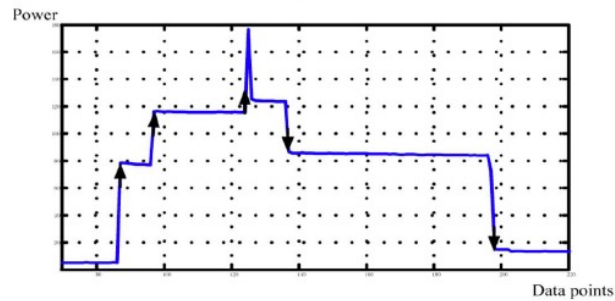
- Sequence signature describes the logical sequence of operation events of a load.
- In another word, it represents the sequence of appearances of edges.
- For example, a washer usually follows the following operating modes: water-fill, immerse, rinse, drainage, and spin-dry.



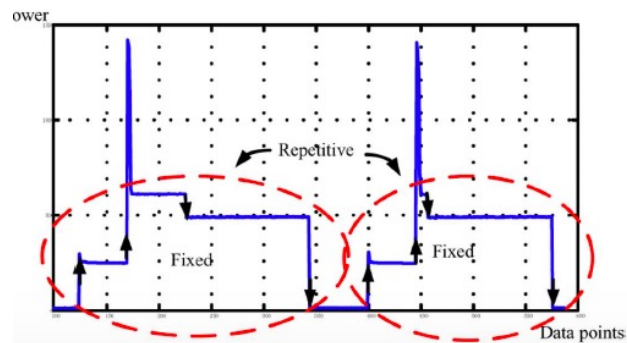
Sequence signature



(a)



(b)



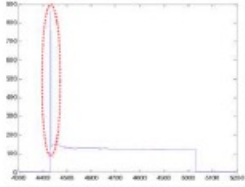
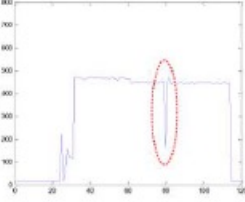
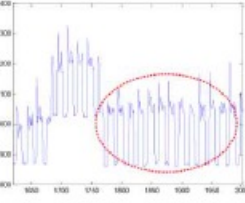
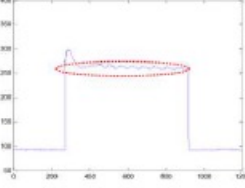
(c)

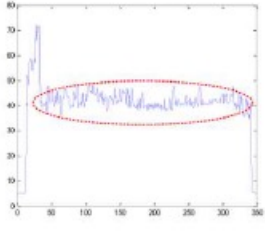
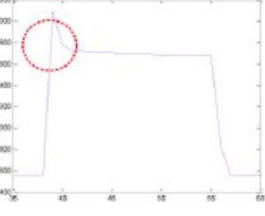
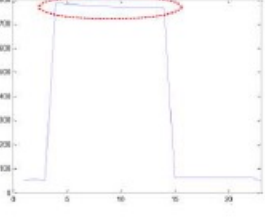


Trend signature

- A trend signature refers to variation of power demand between two edges.
- A TV set may experience a falling spike at moments of switching channels; pulses are usually caused by electronic switches.
- A lot of stoves have pulses because they have an integer-cycle controller in it; it prevents itself from overheating.

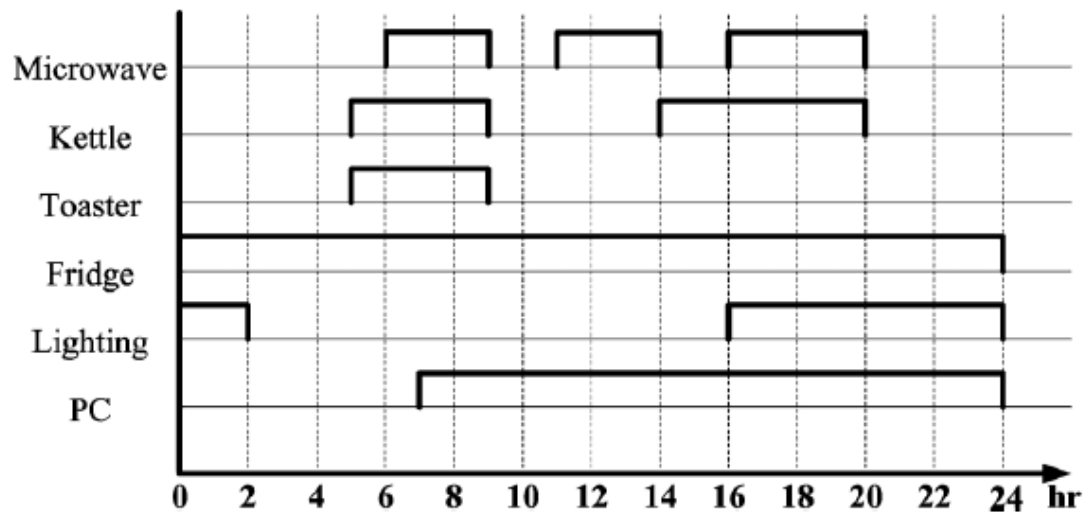
Trend signature

Type	Curve example	Power slope feature
Rising spike		A large negative slope following a larger positive slope
Falling spike		A large positive slope following a large negative slope
Pulses		Continuous pairs of large slopes
Fluctuation		Continuous small slopes; signs of slopes slowly change

Quick vibrate		Continuous small slopes; signs of slopes quickly change
Gradual falling		Continuous small negative slopes
Flat		Continuous small slopes

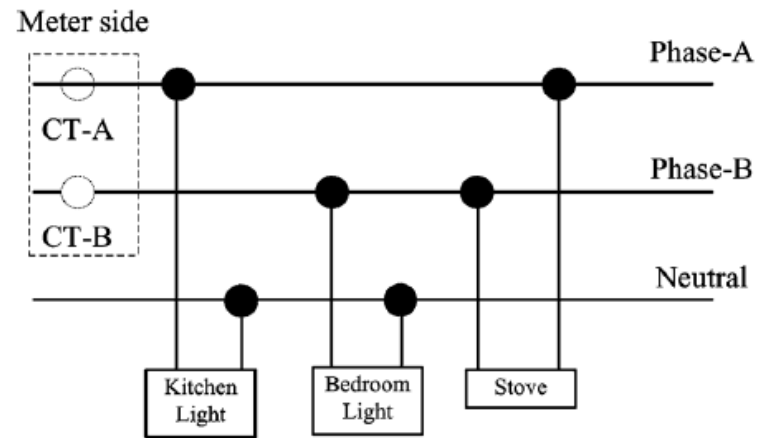
Time/Duration signature

- The time of load window appearance relates close to its function.
- Microwaves are more expected to be seen before breakfast, lunch, and supper
- Lights are usually turned on in the early morning or after dark
- Fridge and furnace are likely to run throughout 24 h.



Phase signature

- There are two 120 V hot wires installed in a typical North American residential house.
- Hereby, the two wires can be named as A and B. Most appliances are connected between A or B and neutral.
- However, some heavy appliances such as stove and dryer are connected between A and B to gain a 240 V voltage.
- We can differentiate from the phase pattern.



Load identification

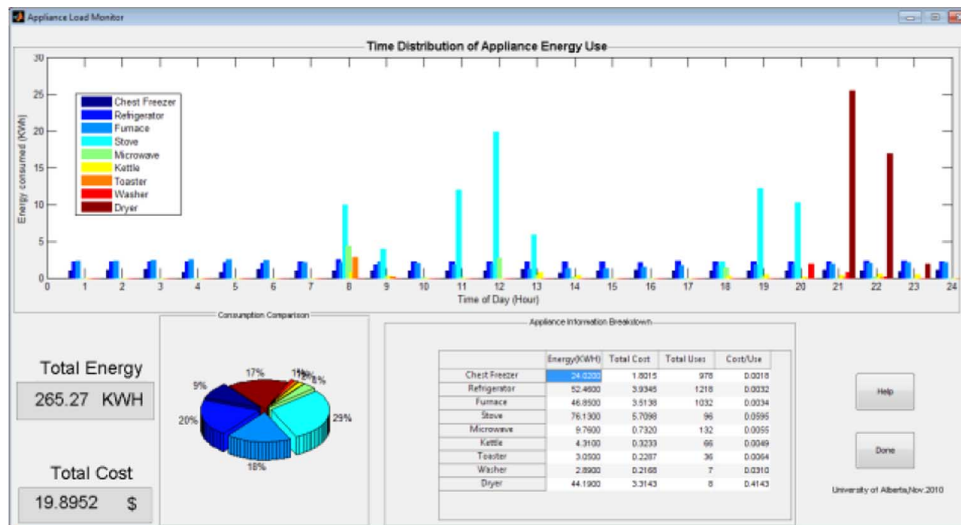
- Inputs x vector
 - S_{edge}
 - S_{seq}
 - S_{trd}
 - S_{time}
 - S_{phase}
- Classifier $G_k(x) = \beta^T x - \gamma_k$
- γ_k is a threshold for type k appliance
- Type k is ON if $G_k(x) > 0$
- β^T and γ_k determined by linear regression and optimized by validation set

Results

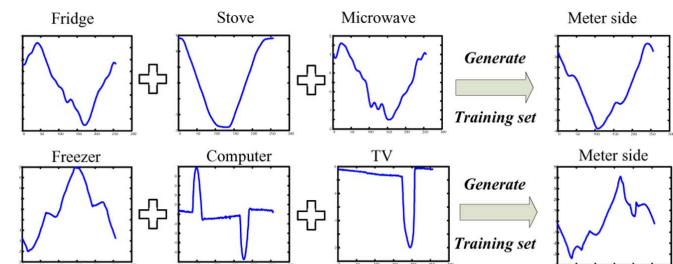
- Weights associated with first four signatures

Load name	Distinctive signatures	ω^T	γ
Fridge	Edge, trend	[0.53 0.17 0.22 0.08]	0.85
Microwave	Edge, time	[0.65 0.19 0 0.16]	0.85
Furnace	Edge, sequence	[0.53 0.47 0 0]	0.85
Stove	Edge, sequence, time	[0.51 0.3 0 0.19]	0.85
Washer	Edge, sequence	[0.55 0.45 0 0]	0.8
Kettle	Edge	[0.86 0.14 0 0]	0.85
Laptop	Edge, trend	[0.5 0.25 0.25 0]	0.8
Average	---	[0.59 0.28 0.07 0.06]	0.85

Verification



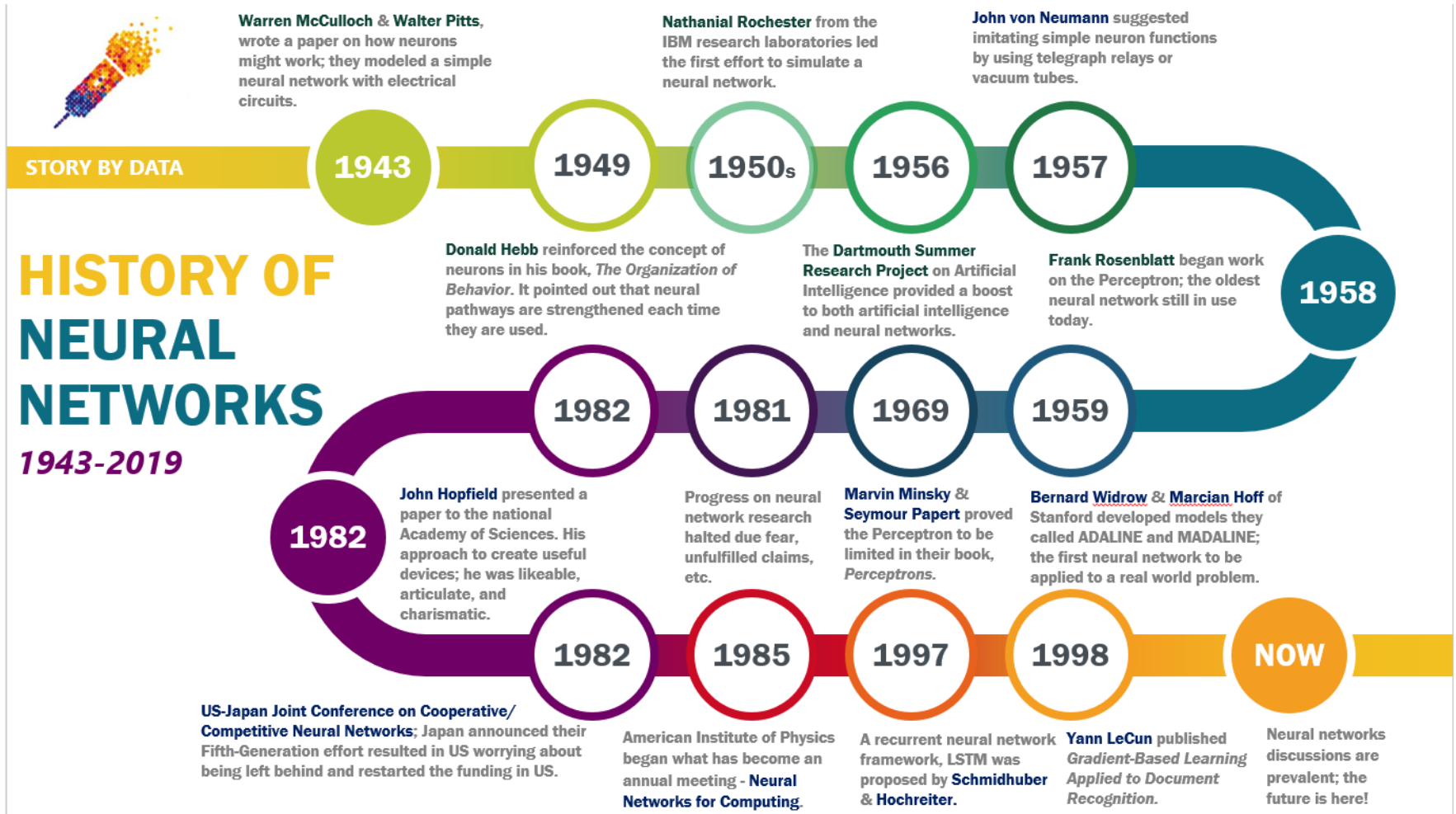
Appliance Name	Actual operation times	Correctly identified operation times	False identified operation times	Identification accuracy(%)
Chest Freezer	178	163	2	90.5
Fridge	213	203	4	93.4
Furnace	185	172	1	92.4
Stove/Oven	16	16	0	100
Microwave	23	22	0	95.7
Kettle	12	11	0	91.6
Toaster	6	6	0	100
Washer	3	3	0	100
Dryer	4	4	0	100



Outline

- Smart meters
- Linear classification
- Neural networks

History



<https://medium.com/analytics-vidhya/brief-history-of-neural-networks-44c2bf72eec>

Basic idea

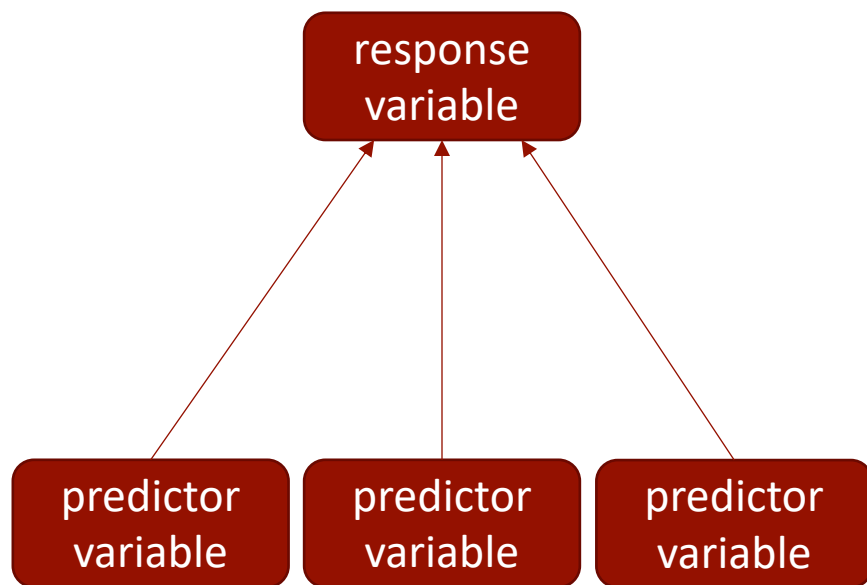
- A class of learning methods that was developed separately in different fields—statistics and artificial intelligence—based on essentially identical models.
- The central idea is to extract linear combinations of the inputs as derived features, and then model the target as a nonlinear function of these features.
- The result is a powerful learning method, with widespread applications in many fields.
- Used in everywhere in smart cities: autonomous driving, intelligent transportation systems, smart grids, urban informatics...

Background

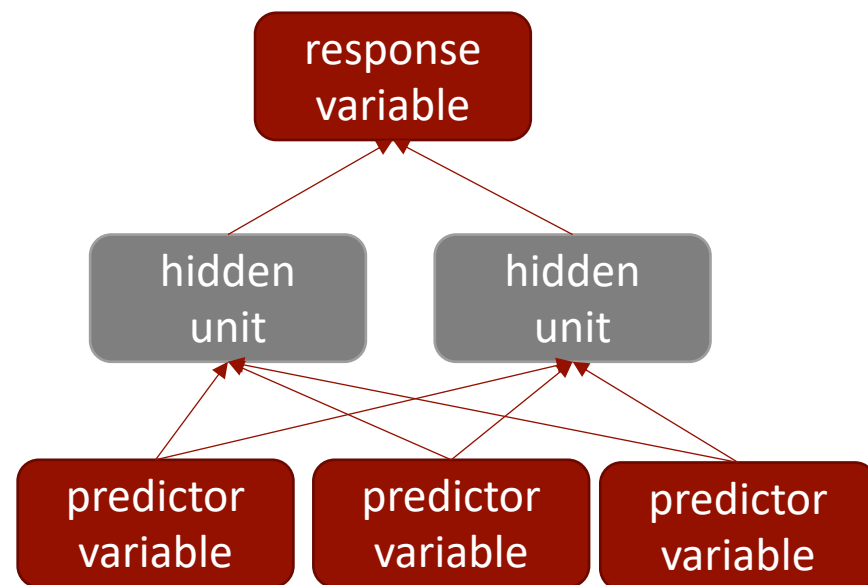
- The term neural network has evolved to encompass a large class of models and learning methods.
- Here we describe the most widely used neural net, sometimes called the single hidden layer back-propagation network, or single layer perceptron.
- There has been a great deal of hype surrounding neural networks, making them seem magical and mysterious.
- As we make clear in this course, they are just nonlinear statistical models.

Introduction

- A two-stage regression or classification model
- Instead of directly feeding predictor variables into a regression/classification function, we put a set of intermediate derived features or hidden units in between.



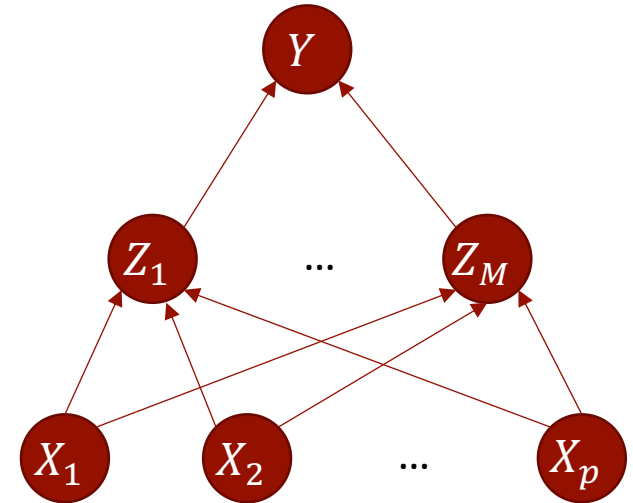
linear regression (LR)



neural network (NN)

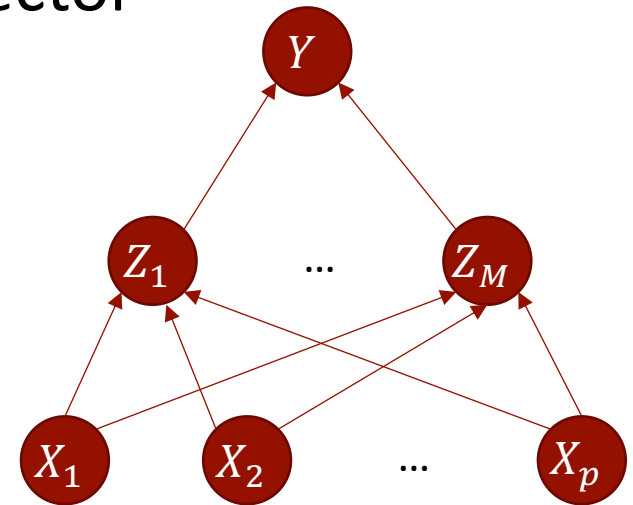
NN regression: 1-dimensional output

- Suppose we have a p -dimensional input vector $X = [X_1 \ X_2 \ \dots \ X_p]^T$
- Our objective is to predict an output scalar Y from X
- Hidden units: an M -dimensional vector $Z = [Z_1 \ Z_2 \ \dots \ Z_M]^T$
- Z is given by the **sigmoid function**:
- $$Z_1 = \frac{1}{1 + \exp(\alpha_{01} + \alpha_1^T X)}$$
- $$Z_2 = \frac{1}{1 + \exp(\alpha_{02} + \alpha_2^T X)}$$
- $$Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$$
- $m = 1, 2, \dots, M$

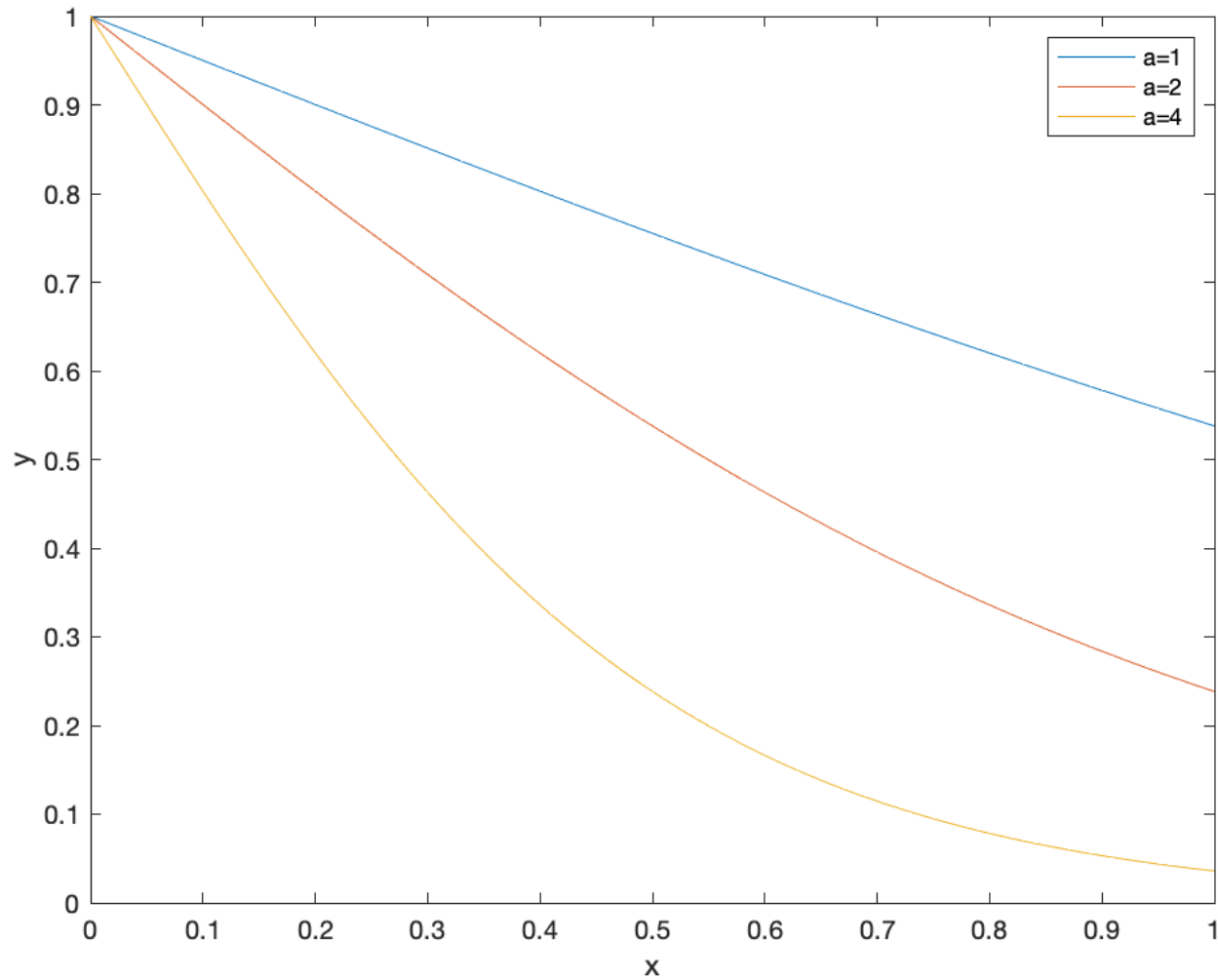


NN regression: 1-dimensional output

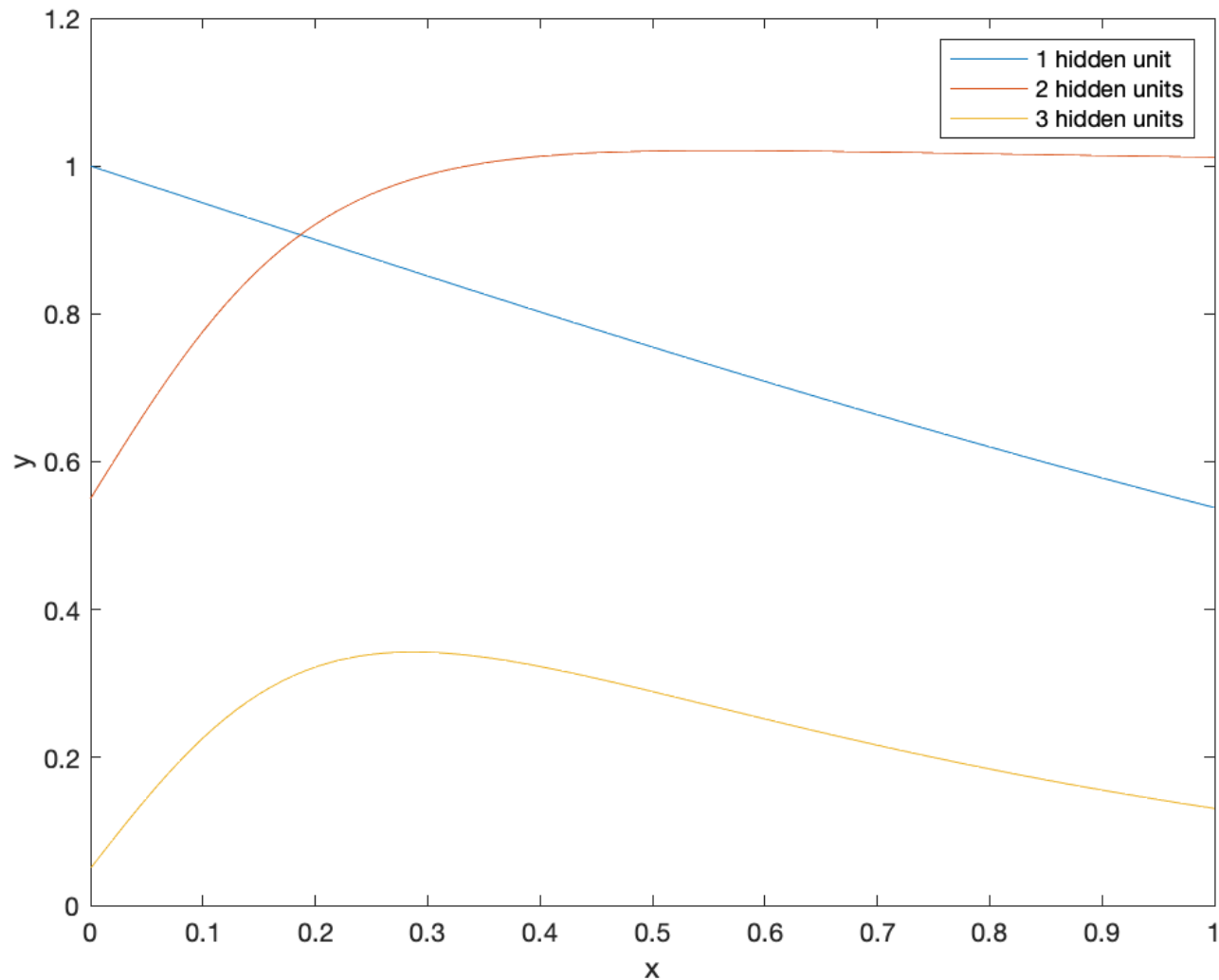
- $Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$, $m = 1, 2, \dots, M$
- α_{0m} is a scalar
- α_m is a p -dimensional vector
- Output $Y = \beta_0 + \beta^T Z$
- β_0 is scalar, β is M -dimensional vector
- Thus, we have constructed a neural network.
- The NN is essentially a nonlinear regression.



Example: 1 input, 1 hidden unit



Example: 1 input, 1--3 hidden units



Hidden units

- The units in the middle of the network, computing the derived features Z_m , are called hidden units because the values Z_m are not directly observed.
- In general there can be more than one hidden layer: **deep neural networks**.
- We can think of the Z_m as a basis expansion of the original inputs X ; the neural network is then a standard linear model, or linear multilogit model, using these transformations as inputs.
- There is, however, an important enhancement over the standard basis expansion techniques discussed; here the parameters of the basis functions are learned from the data.

Hidden units

- General form of hidden unit:

$$Z_m = \sigma(\alpha_{0m} + \alpha_m^T X)$$

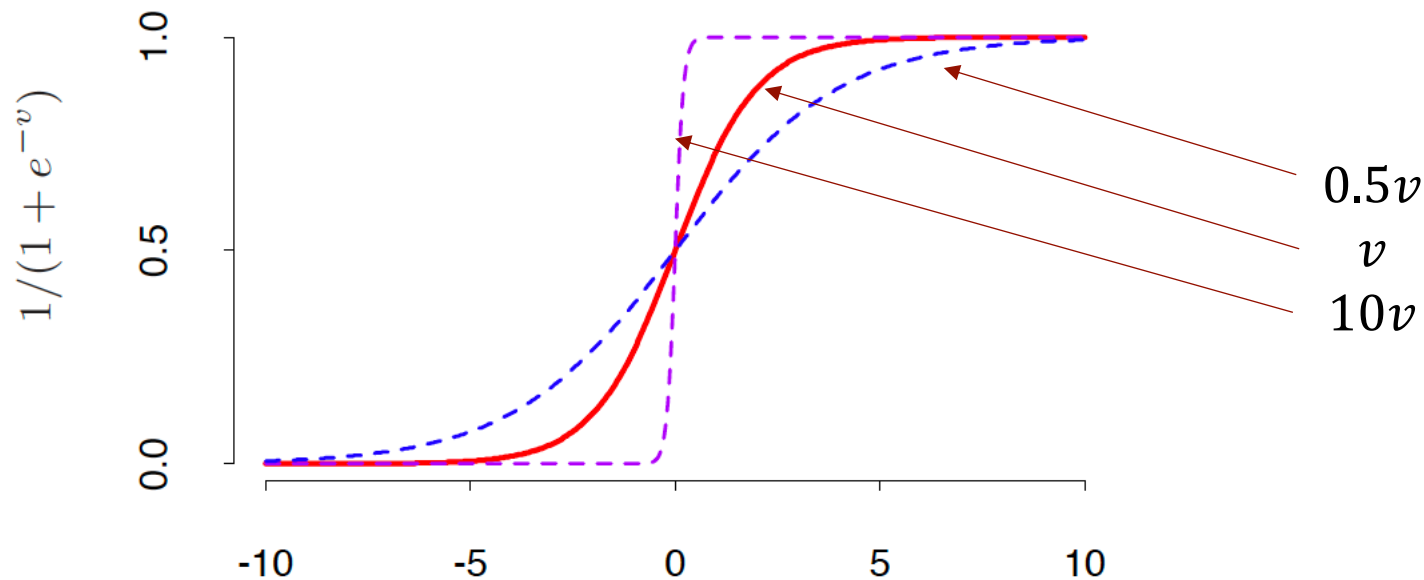
- Notice that if σ is the identity function, then the entire model collapses to a linear model in the inputs.
- Hence a neural network can be thought of as a nonlinear generalization of the linear model, both for regression and classification.
- By introducing the nonlinear transformation σ , it greatly enlarges the class of linear models.
- Typical choice: sigmoid function

$$Z_m = \sigma(\alpha_{0m} + \alpha_m^T X) = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$$

Impact of α_m

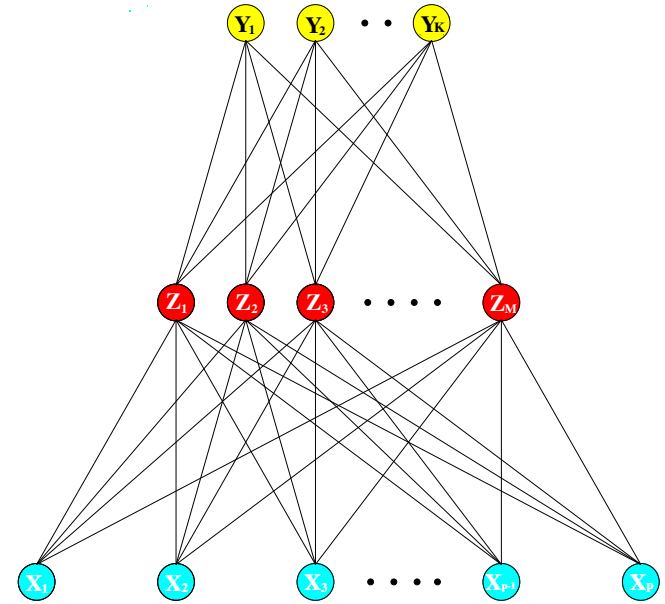
$$Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$$

- The rate of activation of the sigmoid depends on the norm of α_m , and if $\|\alpha_m\|$ is very small, the unit will indeed be operating in the linear part of its activation function.



NN regression: K -dimensional output

- Suppose we have a p -dimensional input vector $X = [X_1 \ X_2 \ \dots \ X_p]^T$
- Our objective is to predict a K -dimensional output vector $Y = [Y_1 \ Y_2 \ \dots \ Y_K]^T$ from X
- Hidden units: an M -dimensional vector $Z = [Z_1 \ Z_2 \ \dots \ Z_M]^T$
- $$Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)},$$
$$m = 1, 2, \dots, M$$
- $$Y_k = \beta_{0k} + \beta_k^T Z, \quad k = 1, \dots, K$$



NN regression: K -class classification

- Suppose that every entry is described by a p -dimensional input vector $X = [X_1 \ X_2 \ \dots \ X_p]^T$, or features.
- Every entry can fall into one out of K classes.
- We want to put every entry into the class that the entry most likely belongs to.
- How to do this? -> Consider this classification as a K -dimensional regression.
- Response Y_k = probability of belonging to class k
- Then follow the k -dimensional regression method to compute Y_k

NN regression: K -class classification

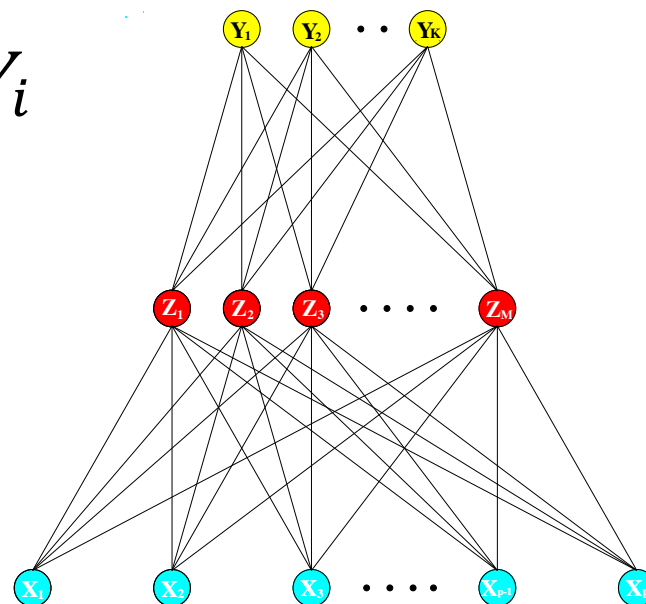
- The regression will give us Y_1, Y_2, \dots, Y_K .
- To compute Y_k , use the softmax formula

$$Y_k = \frac{\exp(\beta_{0k} + \beta_k^T Z)}{\sum_{l=1}^K \exp(\beta_{0l} + \beta_l^T Z)}$$

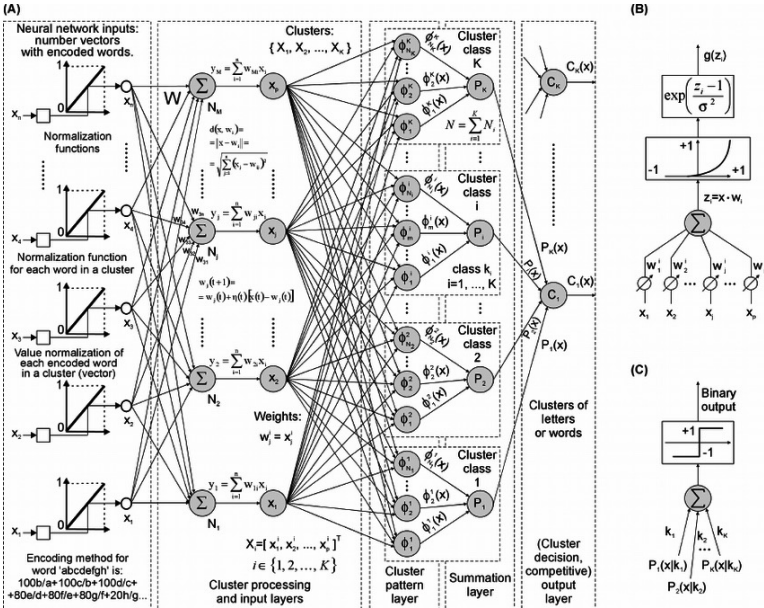
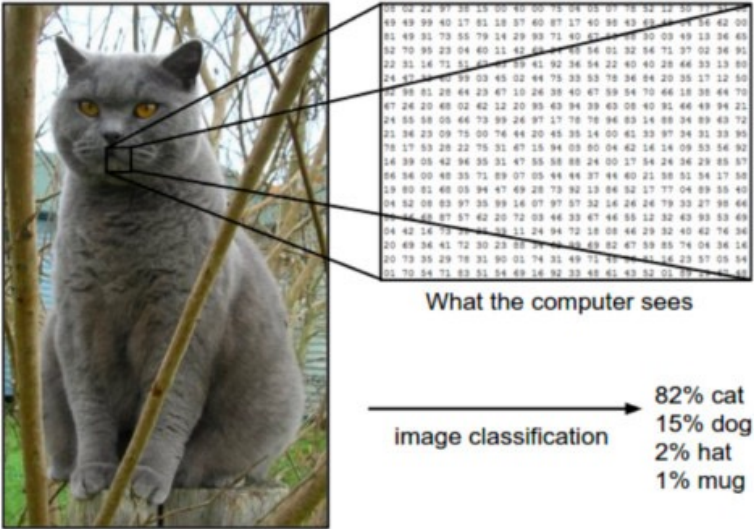
- Find the class k such that

$$Y_k = \max_i Y_i$$

- Put the entry into class k
- -> complete!

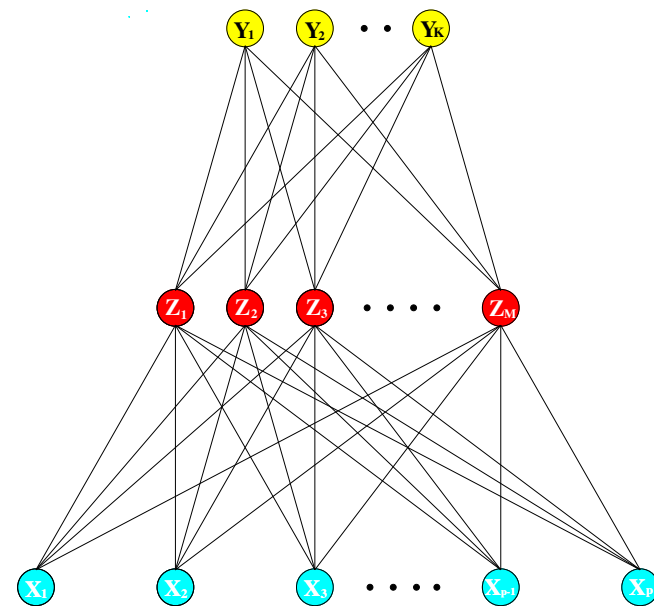
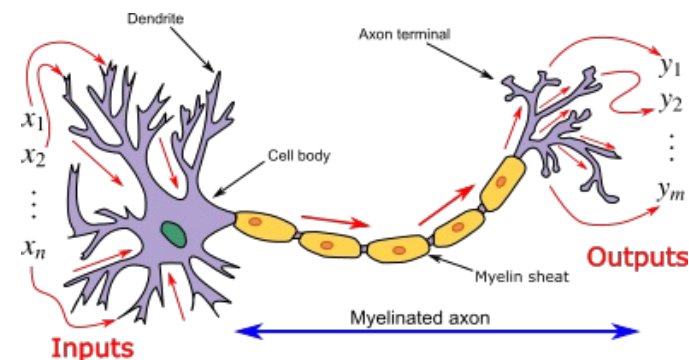


Example: image recognition



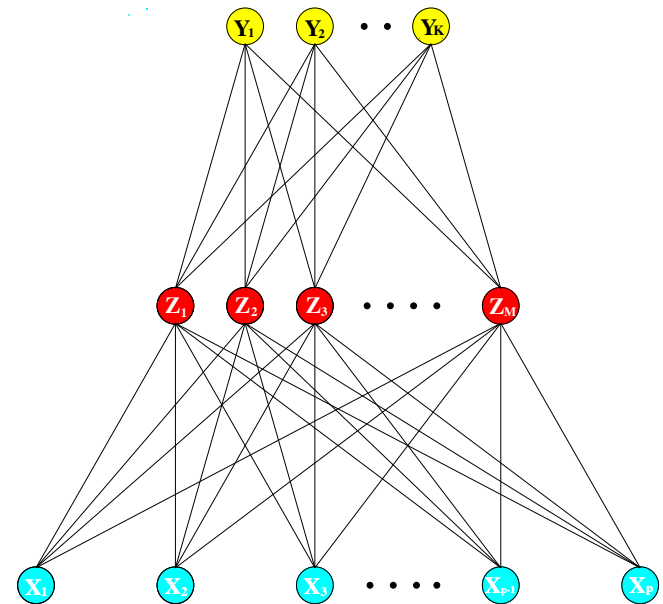
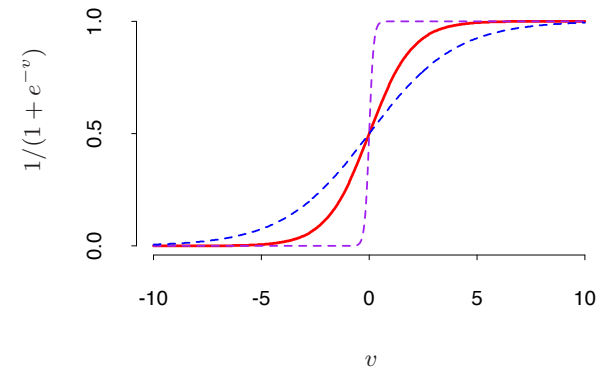
Why “neural” network?

- The model was first derived for human brain
- Each hidden unit represents a neuron
- Each connection represents a synapsis
- Each input represent an external signal (sense)
- A neuron is activated if the total signal passed to it exceed a certain threshold (sigmoid function).



Why “neural” network?

- A neuron is activated if the total signal passed to it exceed a certain threshold (sigmoid function).
- $$Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$$
- Also called activation function
- Turns out to be a useful tool for nonlinear statistical modeling
- To avoid confusion, say “artificial neural network”



Hidden units

- NN is a powerful and very general approach for regression and classification, and has been shown to compete well with the best learning methods on many problems.
- These tools are especially effective in problems with a high signal-to-noise ratio and settings where prediction without interpretation is the goal.
- They are less effective for problems where the goal is to describe the physical process that generated the data and the roles of individual inputs.
- Each input enters into the model in many places, in a nonlinear fashion.
- Some authors plot a diagram of the estimated weights into each hidden unit, to try to understand the feature that each unit is extracting.

When to use NN?

- If linear models apparently fail
- If you have no idea about the predictor-response relation
- If you do not want to select models from case to case
- If insights are not important
- If wrong predictions do not lead to catastrophic consequences

Can we use NN for the following tasks?

- Detect spam emails
- Medical diagnosis
- Judge criminals
- Approximate aerodynamics of an aircraft
- Image recognition
- Vehicle routing
- Ramp metering
- Electrical power allocation

Training a neural network: 1-dimensional output

- For regression model, the hidden units are given by

$$Z_m = \frac{1}{1 + \exp(\alpha_{0m} + \alpha_m^T X)}$$

and the response is given by

$$Y = \beta_0 + \beta^T Z$$

- We can eliminate Z and express Y as a function of X

$$Y = f(X)$$

- We need to determine $\alpha_{0m}, \alpha_m, \beta_0, \beta$, which are called **weights**
- There are $M(p + 1)$ of α 's and $M + 1$ β 's
- Let $\theta = \{\alpha_{0m}, \alpha_m, \beta_0, \beta\}$, i.e. the complete set of all weights

Training a neural network: 1-dimensional output

- Let $\theta = \{\alpha_{0m}, \alpha_m, \beta_0, \beta\}$, i.e. the complete set of all weights
- Suppose that we have a collection of data, i.e. observed predictors x_1, x_2, \dots, x_N and responses y_1, y_2, \dots, y_N
- Given a set of weights θ , the prediction error of a NN model is

$$R(\theta) = \sum_{i=1}^N (y_i - f(x_i))^2$$

- Fitting or training the NN essentially means finding θ that minimize the prediction error.

Training a neural network: K -dimensional output

- We can express Y_k as a function of X

$$Y_k = f_k(X)$$

- Suppose that we have a collection of data, i.e. observed predictors x_1, x_2, \dots, x_N and responses y_1, y_2, \dots, y_N
- Given a set of weights θ , the prediction error of a NN model is

$$R(\theta) = \sum_{k=1}^K \sum_{i=1}^N (y_{ik} - f_k(x_i))^2$$

- Fitting or training the NN essentially means finding θ that minimize the prediction error.

How to find θ that minimizes prediction error?*

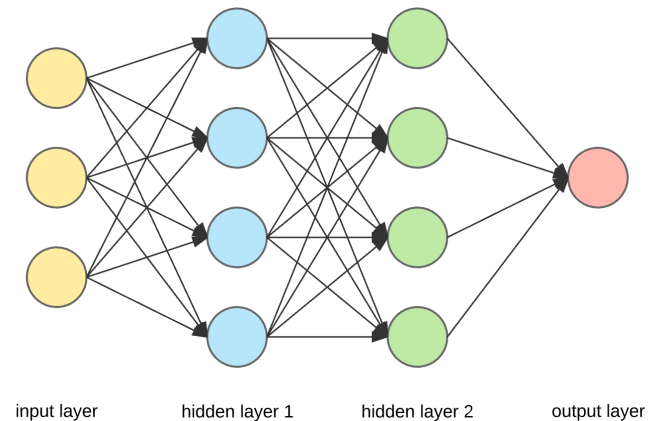
- Generic approach: gradient descent
- Particularly called back-propagation in the NN setting
- That is, start from some initial guess θ_0
- Evaluate the gradient $\nabla_{\theta} R(\theta)$, i.e. the vector of derivatives of $R(\theta)$ w.r.t. every component in θ , at the starting point θ_0
- Then, move some step size in the direction specified by the vector $\nabla_{\theta} R(\theta)$, and arrive at a new guess θ_1
- Compute the gradient at θ_1 and move again
- Keep iterating until no moves can further decrease $R(\theta)$

Implementation

- With the help of computers, you will not have to do the iterations on your own...
- ...unless you are a mathematician proving some properties (e.g. convergence/optimalty) of the gradient descent (i.e. back-propagation) approach.
- You can use Python, R, MATLAB...

Deep neural networks

- Deep = more than one hidden layers
- Can model complex non-linear relationships.
- Compositional models where the object is expressed as a layered composition of inputs.
- The extra layers enable composition of features from lower layers, potentially modeling complex data with fewer units than a similarly performing shallow network.



Train a NN

- Input: vector of x
- Hidden units: vector of $z = g(x; \theta)$
- Output: scalar $y = f(z; \varphi) = f(g(x; \theta); \varphi)$
- $\text{RSS} = \sum (y_i - f(g(x; \theta); \varphi))^2$
- Find parameters θ and φ that minimize RSS
- Need to normalize inputs