Advanced Data Science

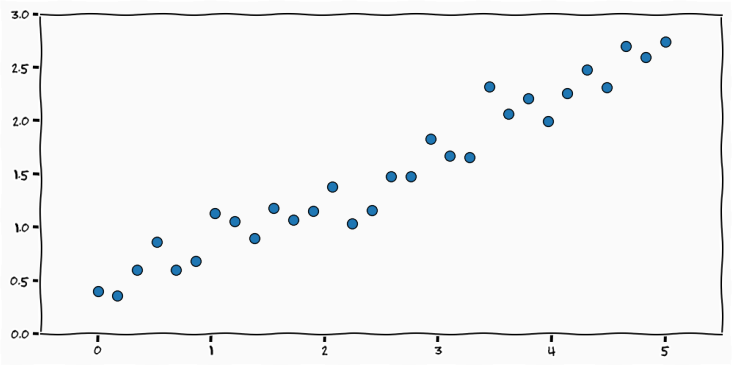
Lecture 8 : Address

Carl Henrik Ek - che29@cam.ac.uk

18th of November, 2024

http://carlhenrik.com

Formalism

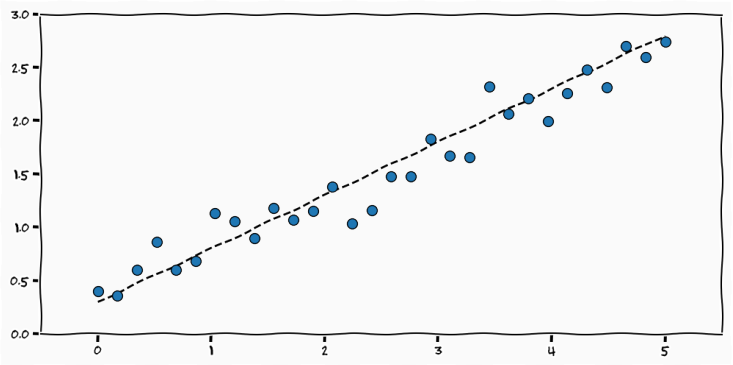


$\mathbf{x} \in \mathcal{X}$ explanatory variable

$y \in \mathcal{Y}$ response variable

Task *explain the response by the explanatory variables*

Linear Regression [Bishop, 2006]



$$y_i = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2).$$

Linear Regression Prediction

$$\mathbb{E}[y_i \mid \mathbf{x}_i] = \mathbb{E}\left[\sum_{j=1}^{d} \beta_j x_{ij} + \epsilon\right]$$

Linear Regression Prediction

$$\mathbb{E}[y_i \mid \mathbf{x}_i] = \mathbb{E}\left[\sum_{j=1}^{d} \beta_j x_{ij} + \epsilon\right]$$
$$= \mathbb{E}\left[\sum_{j=1}^{d} \beta_j x_{ij}\right] + \mathbb{E}[\epsilon]$$

Linear Regression Prediction

$$
\begin{aligned}
\mathbb{E}[y_i \mid \mathbf{x}_i] &= \mathbb{E}\left[\sum_{j=1}^{d} \beta_j x_{ij} + \epsilon\right] \\
&= \mathbb{E}\left[\sum_{j=1}^{d} \beta_j x_{ij}\right] + \mathbb{E}[\epsilon] \\
&= \sum_{j=1}^{d} \beta_j x_{ij} + 0.
\end{aligned}
$$

Linear Regression

$$y_i = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon,$$

Linear Regression

$$y_i = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon,$$

$$y_i + \epsilon = \sum_{j=1}^{d} \beta_j x_{ij},$$

Linear Regression

$$y_i = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon,$$

$$y_i + \epsilon = \sum_{j=1}^{d} \beta_j x_{ij},$$

$$\hat{y}_i = \sum_{j=1}^{d} \beta_j x_{ij},$$

$$y_i = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon,$$

$$y_i + \epsilon = \sum_{j=1}^{d} \beta_j x_{ij},$$

$$\hat{y}_i = \sum_{j=1}^{d} \beta_j x_{ij},$$

$$\hat{y}_i \sim \mathcal{N}(\hat{y}_i \mid y_i, \sigma^2) = \mathcal{N}\left(\hat{y}_i \mid \sum_{j=1}^{d} \beta_j x_{ij}, \sigma^2\right),$$

Generalised Linear Models [McCullagh et al., 1989]

$$g(\mathbb{E}[y_i \mid \mathbf{x}_i]) = \sum_{j=1}^{d} \beta_j x_{ij},$$

$g(\cdot)$ link function

$y \sim \mathcal{D}$ Exponential Dispersion Family

$\sum_{j=1}^{d} \beta_j x_{ij}$ Linear predictor

Generalised Linear Models [McCullagh et al., 1989]

$$\mathbb{E}[y_i \mid \mathbf{x}_i] = g^{-1}\left(\sum_{j=1}^{d} \beta_j x_{ij}\right),$$

- The inverse of the *link* maps the linear predictor to the first moment of the response
- Linear regression the link is identity

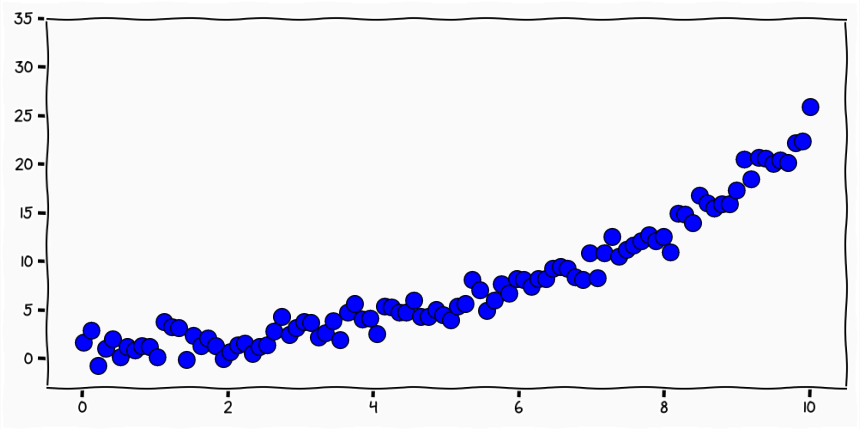
[1] https://towardsdatascience.com/glms-part-iii-deep-neural-networks-as-recursive-generalized-linear-URL

Generalised Linear Models [McCullagh et al., 1989]

$$\mathbb{E}[y_i \mid \mathbf{x}_i] = g^{-1}\left(\sum_{j=1}^{d} \beta_j x_{ij}\right),$$

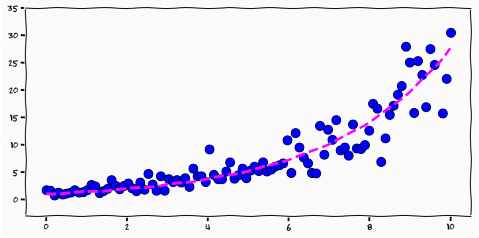
- The inverse of the *link* maps the linear predictor to the first moment of the response
- Linear regression the link is identity
- Looks an awful lot like a neural network[1]

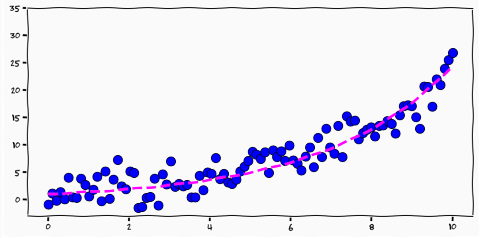
[1] https://towardsdatascience.com/glms-part-iii-deep-neural-networks-as-recursive-generalized-linear-URL



$$y_i = f(x_i) + \epsilon$$
$$\epsilon \sim \mathcal{N}(0, \sigma^2)$$

Transformation vs GLM





$$\log(y_i) = \sum_{j=1}^{d} \beta_j x_{ij} + \epsilon_i$$

$$y_i = e^{\sum_{j=1}^{d} \beta_j x_{ij} + \epsilon_i} = e^{\sum_{j=1}^{d} \beta_j x_{ij}} e^{\epsilon_i}$$

$$\log(\hat{y}_i) = \sum_{j=1}^{d} \beta_j x_{ij}$$

$$y_i = e^{\sum_{j=1}^{d} \beta_j x_{ij}} + \epsilon$$

Exponential Dispersion Family [2]

$$f(y;\theta,\phi) = e^{\frac{\theta y - b(\theta)}{a(\phi)} + c(y,\phi)},$$

θ location parameter

ϕ scale parameter

i.i.d. we will assume that the data is drawn i.i.d.

[2] https://en.wikipedia.org/wiki/Exponential_dispersion_model

Gaussian

$$f(y;\mu,\sigma^2) = e^{\frac{\mu y - \frac{1}{2}\mu^2}{\sigma^2} - \frac{y^2}{2\sigma^2} - \frac{1}{2}\ln(2\pi\sigma^2)}$$

- $\theta = \mu$
- $\phi = \sigma^2$
- $b(\theta) = \frac{1}{2}\mu^2$

- $a(\phi) = \sigma^2$
- $c(y,\phi) = \frac{y^2}{2\sigma^2} - \frac{1}{2}\ln(2\pi\sigma^2)$

$$\mathbb{E}[y \mid \mathbf{x}] = \frac{\partial}{\partial \theta} b(\theta)$$
$$\mathbb{V}[y \mid \mathbf{x}] = a(\phi) \frac{\partial^2}{\partial \theta^2} b(\theta).$$

- Through a consistent parametrisation we can generalise the moment calculations

Statsmodels

Code

```
import statsmodels.api as sm
m = sm.GLM(y, x, sm.families.Gaussian(sm.families.links.log()))
m_r = m.fit()
y_p = m_r.get_prediction(x_p).summary_frame(alpha=0.05)['mean']
```

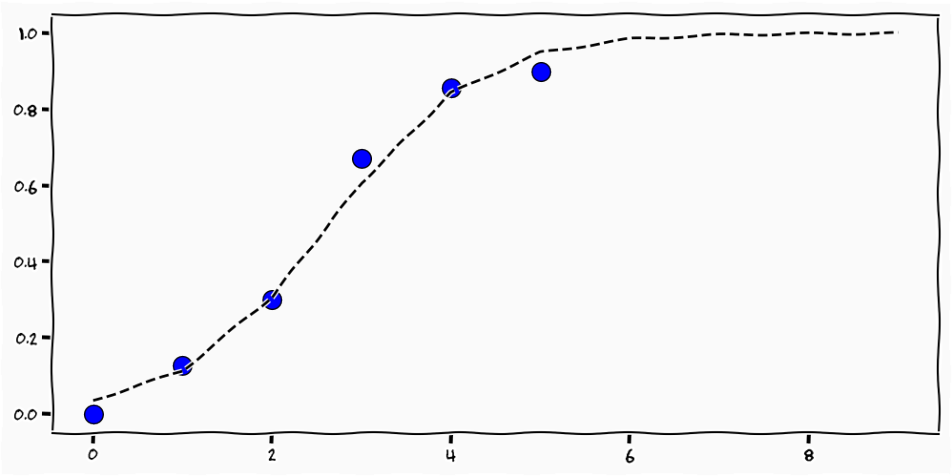
Families `Binomial, Gamma, Gaussian, InverseGaussian, NegativeBinomial, Poisson, Tweedie`

Link Functions `CLogLog, LogLog, Log, Logit, NegativeBinomial, Power, cauchy, identity, inverse_power, inverse_squared, nbinom, probit`

Binomial Data [Weisberg, 2005]

Current	Trials	Response	Proportion
0	70	0	0.00
1	70	9	0.129
2	70	21	0.300
3	70	47	0.671
4	70	60	0.857
5	70	63	0.900

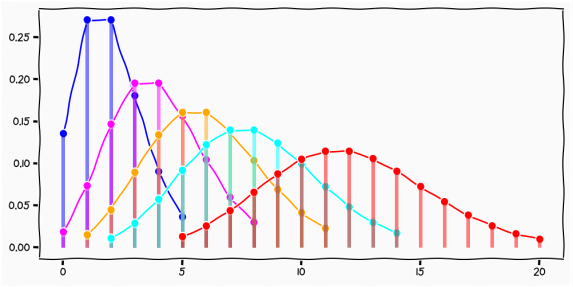
Logistic Regression



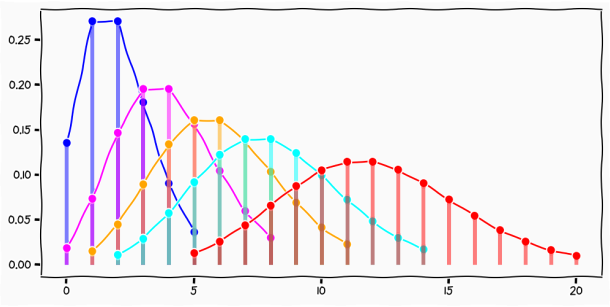

```
sm.GLM(y, X, sm.families.Binomial(sm.families.links.logit()))
```

Poisson Distribution

- Arrival times
- Website visitors
- Job cue for server
- Failures of product



Poisson Distribution



$$\text{Poisson}(y_i) = \frac{e^{-\lambda}\lambda^{y}}{y!}$$

$$\mathbb{E}[y_i] = \lambda$$

Poisson Regression

- Counts are positive so we need a positive link function

$$\log(\lambda_i) = \sum_{j=1}^{d} \beta_j x_{ij}$$

- Counts are positive so we need a positive link function

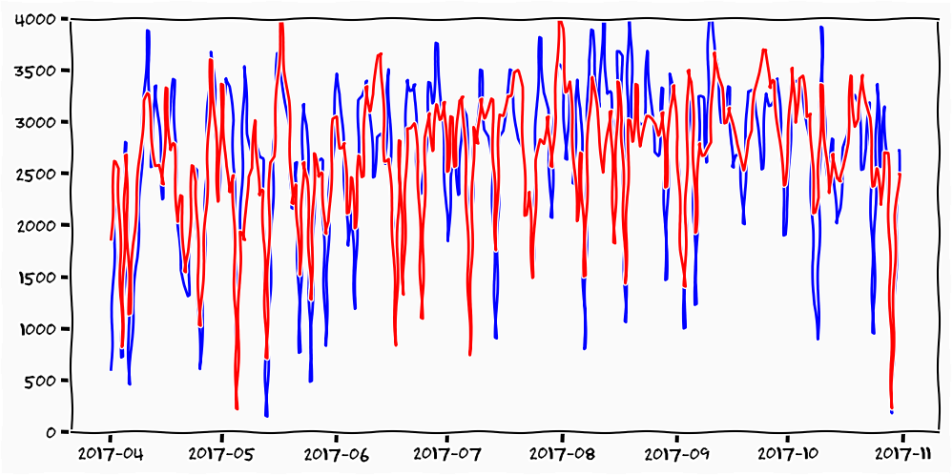
$$\log(\lambda_i) = \sum_{j=1}^{d} \beta_j x_{ij}$$

- Leads to the following model

$$p(y_i \mid x_i) = \frac{e^{-\lambda_i} \lambda_i^{y_i}}{y_i!} = \frac{e^{-\left(e^{\sum_{j=1}^{d} \beta_j x_{ij}}\right)} \left(e^{\sum_{j=1}^{d} \beta_j x_{ij}}\right)^{y_i}}{y_i!}$$

Day	Day of Week	Month	High Temp	Low Temp	Percipitation	Cyclists
1.0	5.0	4.0	46.0	37.0	0.00	606.0
2.0	6.0	4.0	62.1	41.0	0.00	2021.0
3.0	0.0	4.0	63.0	50.0	0.03	2470.0
4.0	1.0	4.0	51.1	46.0	1.18	723.0
6.0	3.0	4.0	48.9	41.0	0.73	461.0
...	...	...	...	...	...	...
1	31	10	54	44	0.00	2727

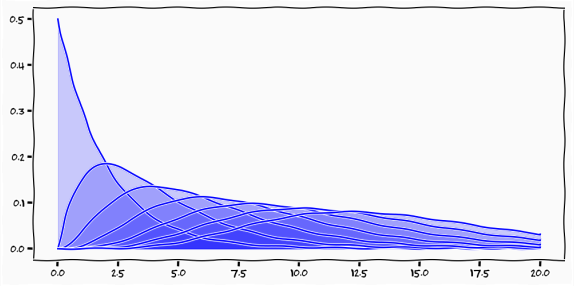
Poisson Regression




```
sm.GLM(y, X, family=sm.families.Poisson()).fit()
```

Gamma Distribution

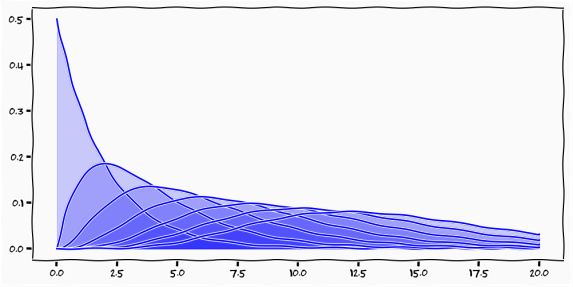
- Waiting times for Poisson events
- Variance and mean connected



Gamma Distribution

$$\text{Gamma}(y_i) = \frac{1}{\Gamma(\phi)\theta^{\phi}} y_i^{\phi-1} e^{-\frac{y_i}{\theta}}$$

$$\mathbb{E}[y_i] = \phi\theta$$
$$\mathbb{V}[y_i] = \phi\theta^2$$



Gamma Regression

- Exponential Dispersion Gamma

$$\text{Gamma}(y_i) = e^{\frac{y_i\theta_i - \log\left(-\frac{1}{\theta_i}\right)}{\phi} + \frac{1-\phi}{\phi}\log(y_i) - \log\left(\Gamma(\phi^{-1}\right)}$$

Gamma Regression

- Exponential Dispersion Gamma

$$\text{Gamma}(y_i) = e^{\frac{y_i\theta_i - \log\left(-\frac{1}{\theta_i}\right)}{\phi} + \frac{1-\phi}{\phi}\log(y_i) - \log\left(\Gamma(\phi^{-1}\right)}$$

- If you "derive" the canonical link function from the distribution it should be,

$$-\frac{1}{\theta} = \sum_{j=1}^{d} \beta_j x_{ij}$$

Gamma Regression

- Exponential Dispersion Gamma

$$\text{Gamma}(y_i) = e^{\frac{y_i\theta_i - \log\left(-\frac{1}{\theta_i}\right)}{\phi} + \frac{1-\phi}{\phi}\log(y_i) - \log\left(\Gamma(\phi^{-1})\right)}$$

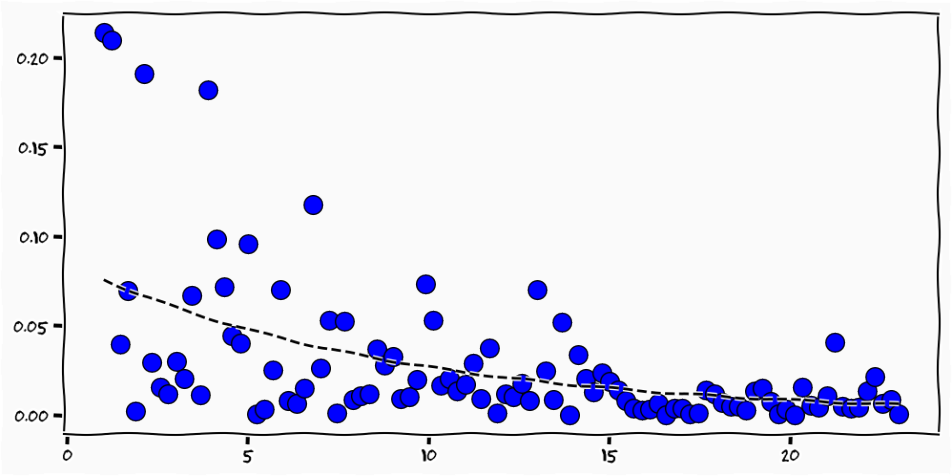
- If you "derive" the canonical link function from the distribution it should be,

$$-\frac{1}{\theta} = \sum_{j=1}^{d} \beta_j x_{ij}$$

- Gamma regression is most commonly used with `log` as the link

$$\log(\mathbb{E}[y_i \mid \mathbf{x}_i]) = \sum_{j=1}^{d} \beta_j x_{ij}$$

Gamma Regression



Model	Response Variable	Link	Explanatory Variable
Linear Regression	Normal	Identity	Continuous
Logistic Regression	Binomial	Logit	Mixed
Poisson Regression	Poisson	Log	Mixed
ANOVA	Normal	Identity	Categorical
ANCOVA	Normal	Identity	Mixed
Loglinear	Poisson	Log	Categorical
Multinomial response	Multinomial	Generalized Logit	Mixed

Inference

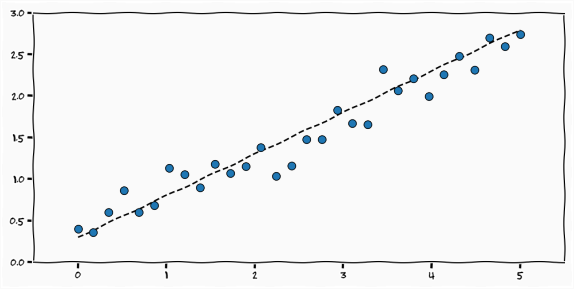
$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{N} p(y_i \mid \boldsymbol{\beta}, \mathbf{x}_i)$$

- In general gradient descent on `log-likelihood`
- For specific models there are tailored inference schemes

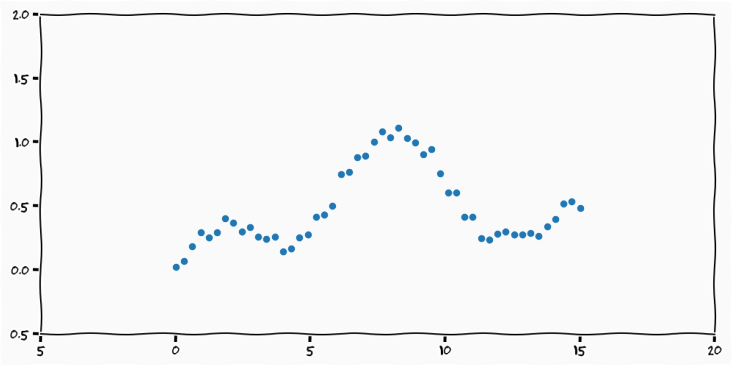
Design Matrix

Design Matrix

$$\mathbf{X} = \begin{bmatrix} x_0 & 1 \\ x_1 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix}$$



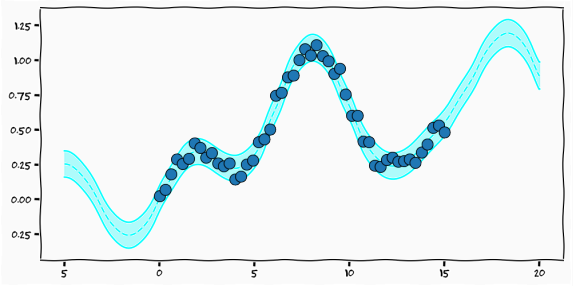
Non-Linear Function



$$y = 0.2\sin(x) + \sin\left(\frac{x^2}{40}\right) + 0.05x$$

Design Matrix

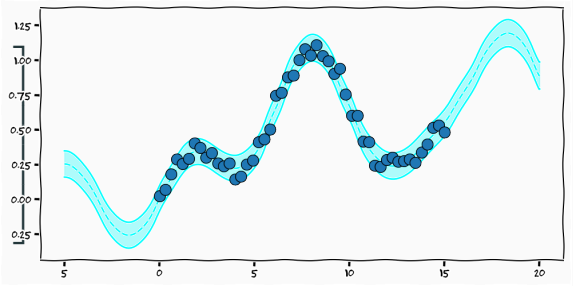
$$\mathbf{X} = \begin{bmatrix} \sin(x_0) & \sin(\frac{x_0^2}{40}) & x_0 \\ \sin(x_1) & \sin(\frac{x_1^2}{40}) & x_1 \\ \vdots & \vdots & \vdots \\ \sin(x_N) & \sin(\frac{x_N^2}{40}) & x_N \end{bmatrix}$$



$$\boldsymbol{\beta} = [0.2155, 0.4956, 0.0482]$$

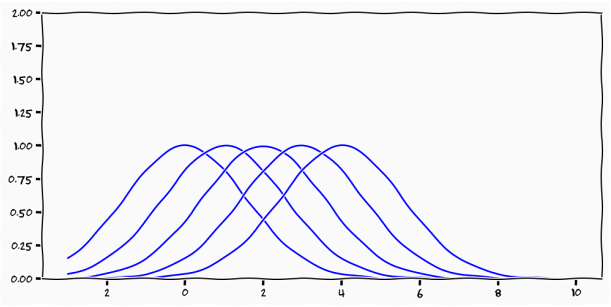
Over-parametrised matrix

$$\begin{bmatrix} \sin(x_0) & \sin(\frac{x_0^2}{40}) & x_0 & -\sin(x_0) \\ \sin(x_1) & \sin(\frac{x_1^2}{40}) & x_1 & -\sin(x_1) \\ \vdots & \vdots & \vdots & \vdots \\ \sin(x_N) & \sin(\frac{x_N^2}{40}) & x_N & -\sin(x_N) \end{bmatrix}$$



$$\boldsymbol{\beta} = [0.1078, 0.4956, 0.0482, 0.1078]$$

Localised Basis Function



$$g\left(\mathbb{E}[y_i \mid \mathbf{x}_i]\right) = \sum_{j=1}^{N} \beta_j \phi(\mathbf{x}_j, \mathbf{x}_i), \qquad \phi(\mathbf{x}_j, \mathbf{x}_i) = e^{-\frac{(\mathbf{x}_i - \mathbf{x}_j)^{\mathrm{T}}(\mathbf{x}_i - \mathbf{x}_j)}{\ell^2}}$$

Solution bias

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{N} p(y_i \mid \boldsymbol{\beta}, \mathbf{x}_i)$$

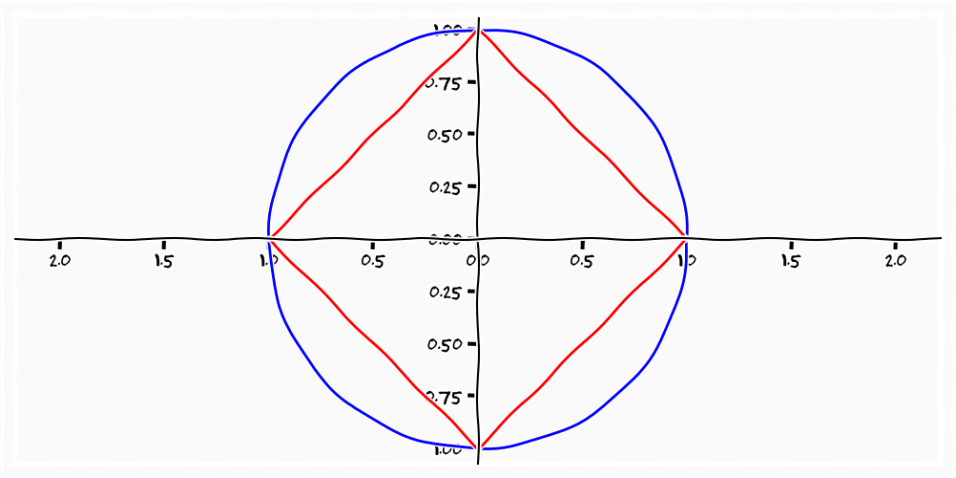
- Maximum Likelihood encodes no preference towards any solution
- Due to optimisation procedure we might get very different results

Norm

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta}}{\operatorname{argmax}} \prod_{i=1}^{N} p(y_i \mid \boldsymbol{\beta}, \mathbf{x}_i) + \lambda \left(\sum_{j=1}^{d} \beta_j^p \right)^{\frac{1}{p}}$$

- Introduce inductive bias towards specific solutions
- Normally done using a norm

Ridge vs Lasso



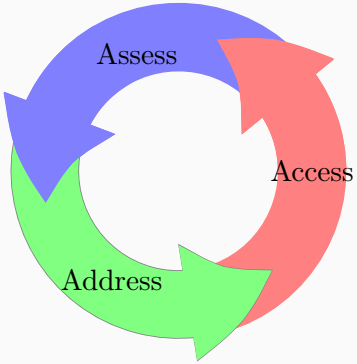
Code

```
m.fit_regularized(alpha=0.10,L1_wt=0.0)
```

- `L1_wt` 0 → `Ridge`, 1 → `Lasso`
- `alpha` the penalty

Address

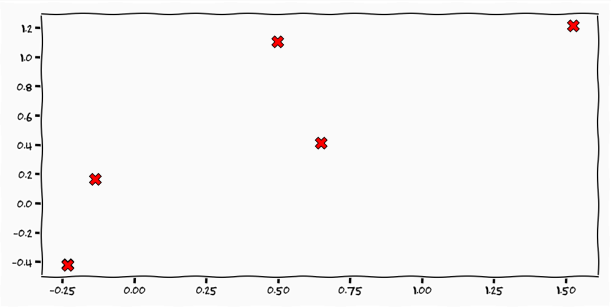
The Datascience Loop



What does Machine Learning do?

$$p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \theta)p(\theta)}{p(\mathcal{D}}$$

Machine Learning Paradigms

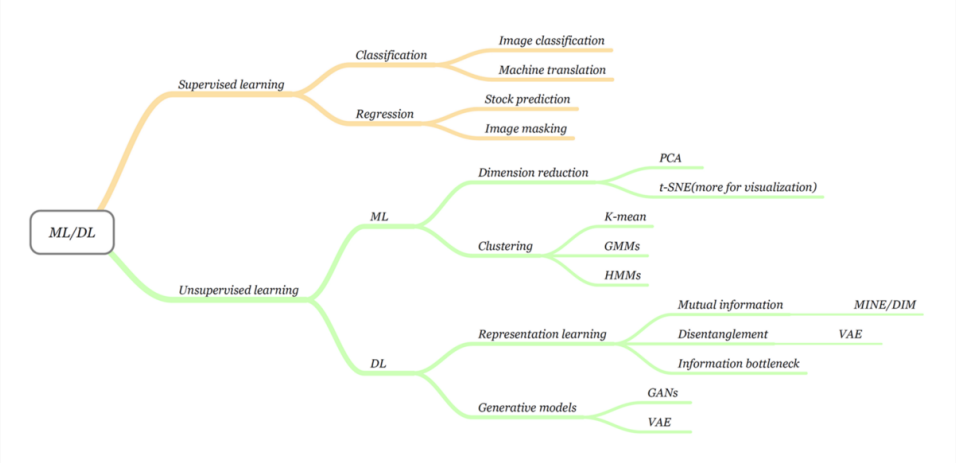


Supervised Learning $p(y \mid x)$

"Unsupervised" Learning $p(y)$

Reinforcement Learning $p(\pi, f \mid \mathcal{L})$

Machine Learning Methods



Domain Set $\mathcal{X}$ the set of measurements/objects that we want to label (input)

Formalism [Shalev-Shwartz et al., 2014]

Domain Set $\mathcal{X}$ the set of measurements/objects that we want to label (input)

Label Set $\mathcal{Y}$ the set of outputs

Formalism [Shalev-Shwartz et al., 2014]

Domain Set $\mathcal{X}$ the set of measurements/objects that we want to label (input)

Label Set $\mathcal{Y}$ the set of outputs

Training Data $\mathcal{S}$ a finite sequence of pairs in $\mathcal{X} \times \mathcal{Y}$

Data Distribution $\mathcal{D}$ probability distribution governing the measurements

Formalism [Shalev-Shwartz et al., 2014]

Data Distribution $\mathcal{D}$ probability distribution governing the measurements

Data Generation $f : \mathcal{X} \to \mathcal{Y}$ the underlying generating process that we wish to recover

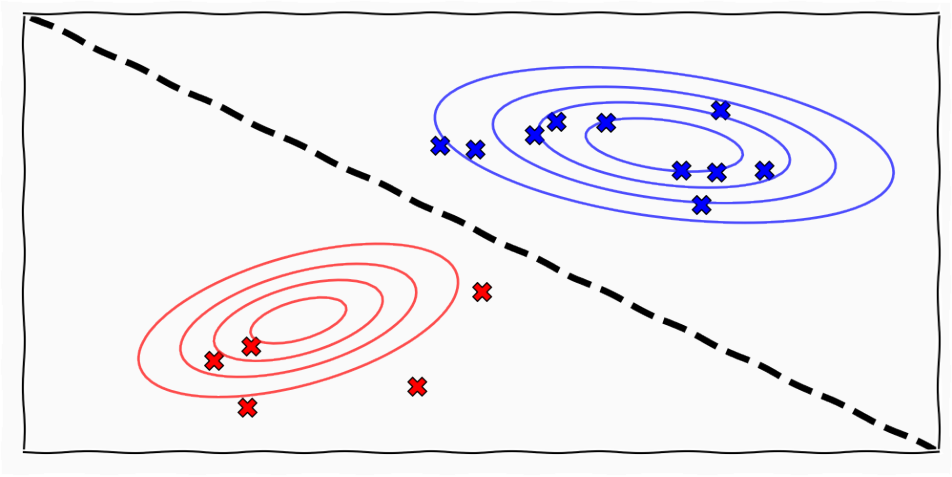
Formalism [Shalev-Shwartz et al., 2014]

Data Distribution $\mathcal{D}$ probability distribution governing the measurements

Data Generation $f : \mathcal{X} \to \mathcal{Y}$ the underlying generating process that we wish to recover

Prediction Rule $h : \mathcal{X} \to \mathcal{Y}$ what we wish to recover, the object that encodes the recovered knowledge

Classification



Measure of Success

$$L_{\mathcal{D},f}(h) := \mathcal{D}(\{x : h(x) \neq f(x)\})$$

- measure of success as probability of misclassified points (true risk)

Measure of Success

$$L_{\mathcal{D},f}(h) := \mathcal{D}(\{x : h(x) \neq f(x)\})$$

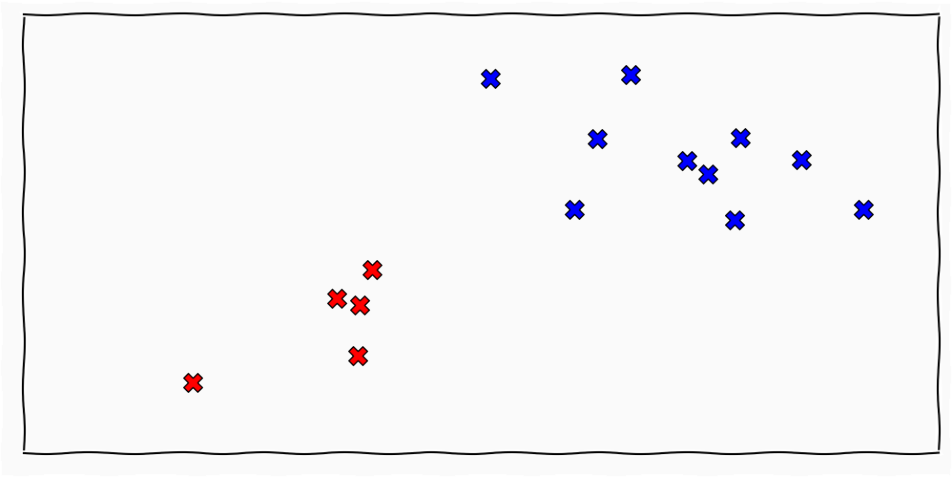
- measure of success as probability of misclassified points (true risk)
- we do not have access to $\mathcal{D}$

Measure of Success

$$L_{\mathcal{D},f}(h) := \mathcal{D}(\{x : h(x) \neq f(x)\})$$

- measure of success as probability of misclassified points (true risk)
- we do not have access to $\mathcal{D}$
- we do not have access to f

Classification

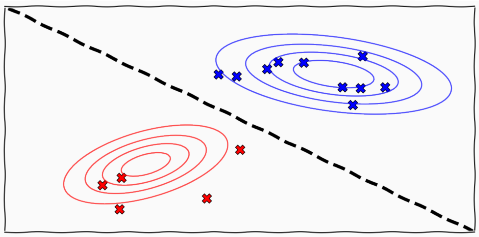


Empirical Risk Minimisation

$$L_{\mathcal{S}}(h) := \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$$

- We assume that $\mathcal{S} \sim \mathcal{D}$
- Empirical measure of risk

Overfitting



$$\mathcal{D} = \frac{1}{3}\mathcal{N}(\cdot,\cdot) + \frac{2}{3}\mathcal{N}(\cdot,\cdot)$$

$$h_{\mathcal{S}}(x) = \begin{cases} y_i & \text{if } \exists i \in [m] \text{s.t. } x_i = x \\ \text{red} & \text{otherwise} \end{cases}$$

- $L_{\mathcal{S}}(h_{\mathcal{S}}) = 0$ for all training data-sets
- if label 0 corresponds to red $L_{\mathcal{D}}(h_{\mathcal{S}}) = \frac{1}{3}$

Algorithm

$$L_{\mathcal{S}}(A(\mathcal{S})) := \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m}$$

- We use an algorithm $A : \mathcal{S} \to h$ to find a hypothesis

$$h_{\mathcal{S}} \in \operatorname*{argmin}_{h \in \mathcal{H}} L_{\mathcal{S}}(h)$$

- We cannot parametrise all possible hypothesis

$\mathcal{A}$ my learning algorithm

The Components of a Learning System

$\mathcal{A}$ my learning algorithm

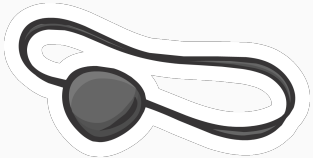
$\mathcal{H}$ my hypothesis class

The Components of a Learning System

$\mathcal{A}$ my learning algorithm

$\mathcal{H}$ my hypothesis class

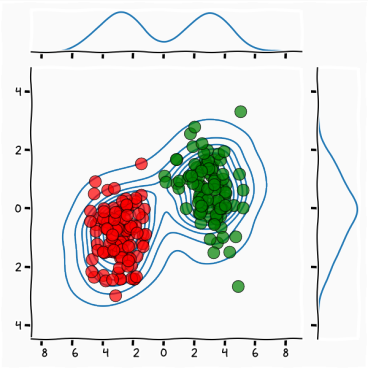
$\mathcal{S}$ my finite trainingset



Statistical Learning

$\mathcal{A}_{\mathcal{H}}(\mathcal{S})$

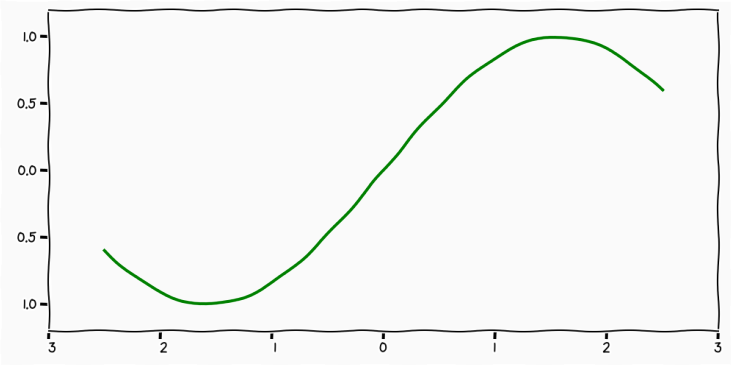
Assumptions: Biased Sample



Statistical Learning

$$\mathcal{A}_{\mathcal{H}}(\mathcal{S})$$

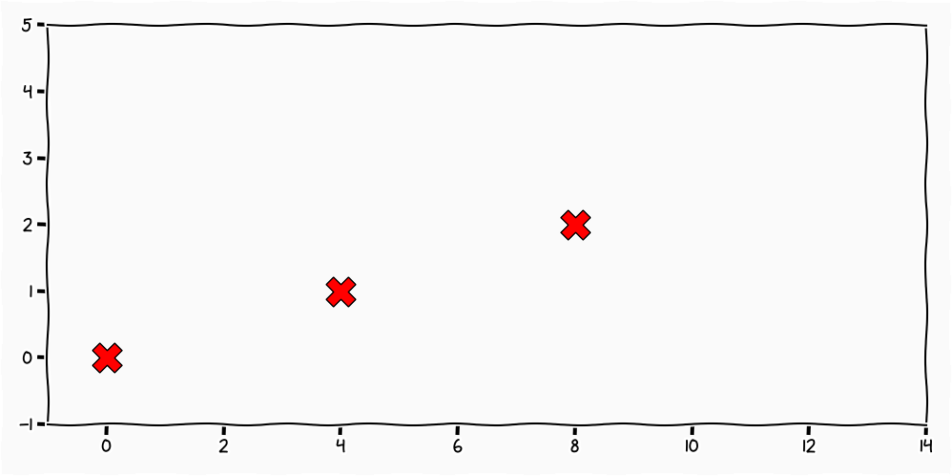
Assumptions: Hypothesis space



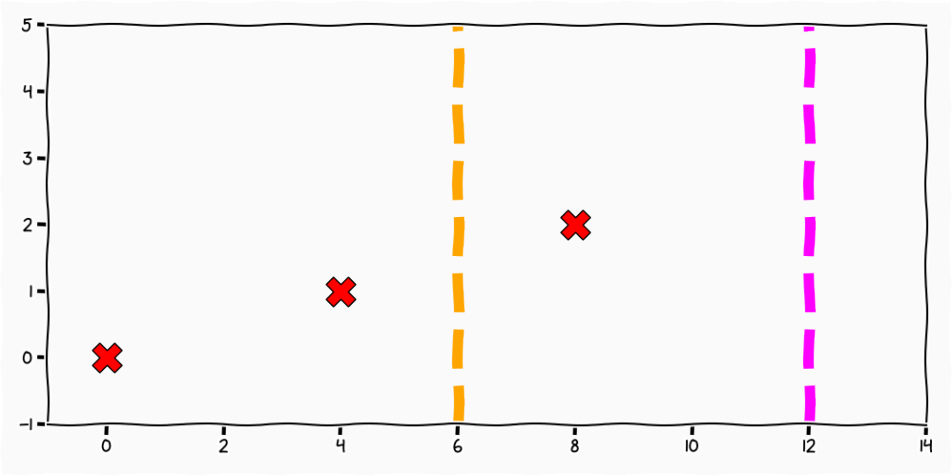
Statistical Learning

$$\mathcal{A}_{\mathcal{H}}(\mathcal{S})$$

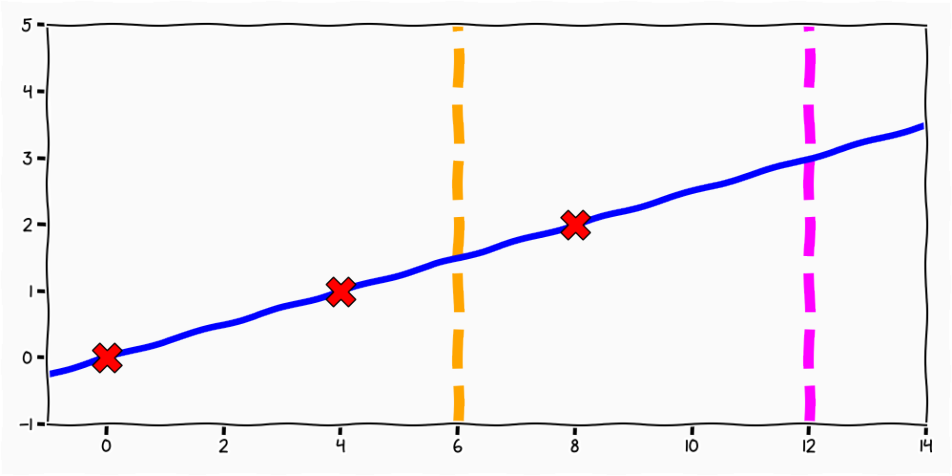
Regression



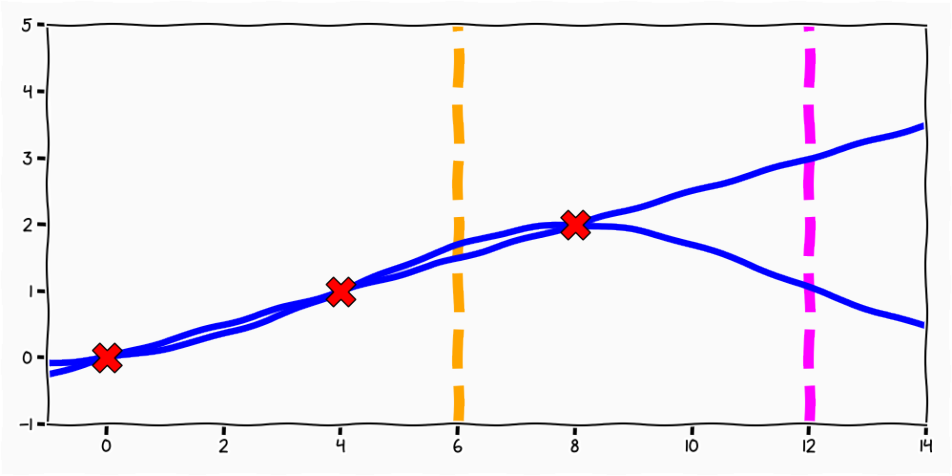
Regression



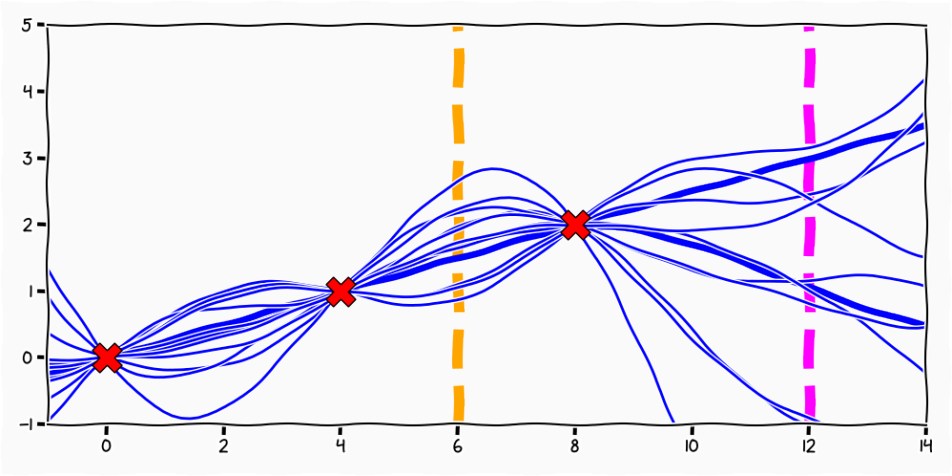
Regression



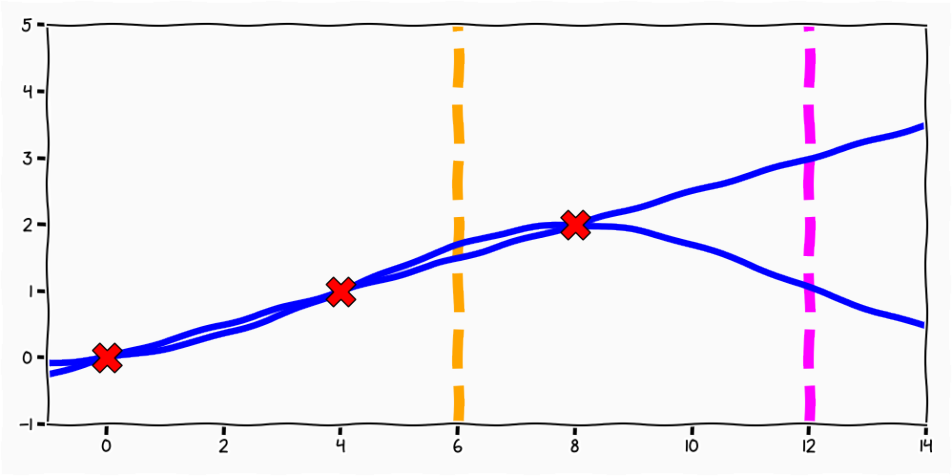
Regression



Regression



Regression



The No-Free Lunch Theorem

- Every algorithm that learns something useful does so by making assumptions

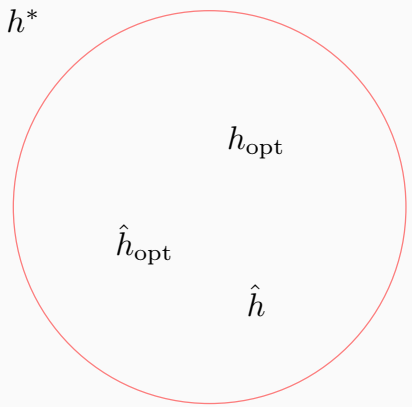
The No-Free Lunch Theorem

- Every algorithm that learns something useful does so by making assumptions
- There exists no universal learner/method/algorithm

The No-Free Lunch Theorem

- Every algorithm that learns something useful does so by making assumptions
- There exists no universal learner/method/algorithm
- There is no free lunch algorithm

The Error Decomposition



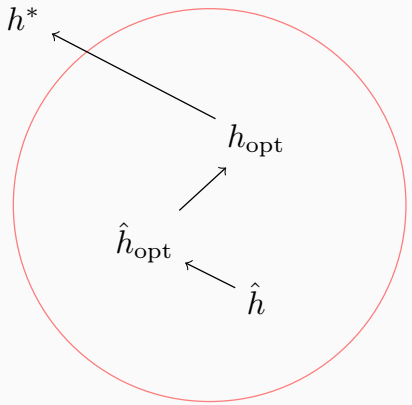
h^* the optimal predictor

$h_{\textbf{opt}}$ the optimal hypothesis

$\hat{h}_{\textbf{opt}}$ the optimal hypothesis on training data

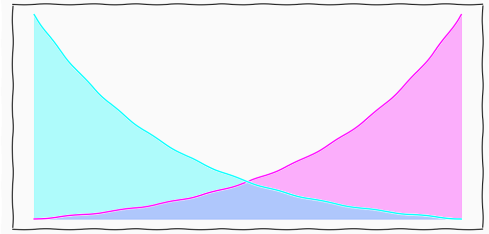
$\hat{h}$ the hypothesis found by learning algorithm

The Error Decomposition



$$
\begin{aligned}
&\epsilon(\hat{h}) - \epsilon(h^*) \\
&= \underbrace{\epsilon(h_{\text{opt}}) - \epsilon(h^*)}_{\text{Approximation}} \\
&+ \underbrace{\epsilon(\hat{h}_{\text{opt}}) - \epsilon(h_{\text{opt}})}_{\text{Estimation}} \\
&+ \underbrace{\epsilon(\hat{h}) - \epsilon(\hat{h}_{\text{opt}})}_{\text{Optimisation}}
\end{aligned}
$$

The Bias-Complexity Trade-off



High Complexity low bias (ϵ_{app} small), but high risk of overfitting (ϵ_{est} large)

Low Complexity high bias (ϵ_{app} large), low risk of overfitting (ϵ_{est} small)

Universal Learner

The No-Free Lunch Theorem

- There exists no universal learner

The No-Free Lunch Theorem

- There exists no universal learner
- For every learner there exist a task on which it fails

The No-Free Lunch Theorem

- There exists no universal learner
- For every learner there exist a task on which it fails
- Every algorithm that learns something useful does so by assumptions

The No-Free Lunch Theorem

- There exists no universal learner
- For every learner there exist a task on which it fails
- Every algorithm that learns something useful does so by assumptions
- *There is no free lunch algorithm*

Statistical Learning Summary

- We can never have sufficient data

Statistical Learning Summary

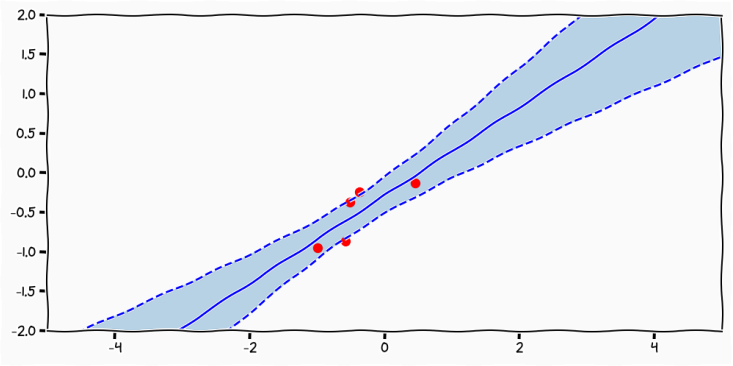
- We can never have sufficient data
- We can never find a method that will guarantee to find the right solution

Statistical Learning Summary

- We can never have sufficient data
- We can never find a method that will guarantee to find the right solution
- We can never be certain about the true risk of our outcome

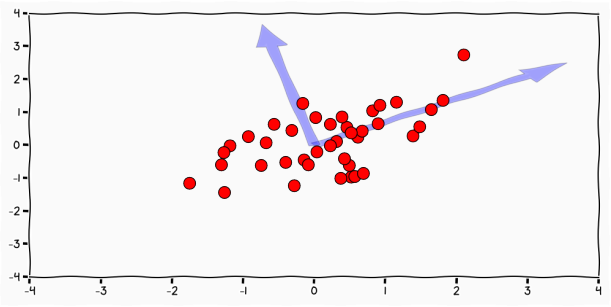
Explicit vs. Tacit Knowledge

Least Squares Regression



Legendre (1805) algorithm that reduces "error"

Gauss (1809) statistical model assuming i.i.d. Gaussian noise



Spearman (1904) proposed an algorithm to extract "factors" from data
Spearman, 1904

Hotelling (1936) concept of factor is clearly defined through a statistical model Hotelling, 1933

Summary

Response Variable Distribution How is your response variable distributed?

Response Variable Distribution How is your response variable distributed?

Link Function How is the scale parameter of the distribution related to the explanatory variables

Response Variable Distribution How is your response variable distributed?

Link Function How is the scale parameter of the distribution related to the explanatory variables

Design Matrix What is the features of the explanatory variables?

Response Variable Distribution How is your response variable distributed?

Link Function How is the scale parameter of the distribution related to the explanatory variables

Design Matrix What is the features of the explanatory variables?

Regulariser What is the "preferred" solution?

Thoughts

- Can you split up the data by some criterion?

Thoughts

- Can you split up the data by some criterion?
  - localised GLM

Thoughts

- Can you split up the data by some criterion?
  - localised GLM
- Can you remove the effect of one model from data and then retrain on residual?

Thoughts

- Can you split up the data by some criterion?
  - localised GLM
- Can you remove the effect of one model from data and then retrain on residual?
- You will not be able to find the "perfect" model, but show that you can reason about these models!!

- the no free lunch cannot be overcome

- the no free lunch cannot be overcome
- explanations are relative to assumptions

- the no free lunch cannot be overcome
- explanations are relative to assumptions
- use models where your assumptions can be made explicit

eof

References

- Bishop, Christopher M. (2006). ***Pattern Recognition and Machine Learning (Information Science and Statistics)***. Secaucus, NJ, USA: Springer-Verlag New York, Inc.
- Hotelling, H (Sept. 1933). "**Analysis of a complex of statistical variables into principal components..**" In: *Journal of Educational Psychology* 24.6, pp. 417–441.
- McCullagh, P. and J. A. Nelder (1989). ***Generalized Linear Models***. London, UK: Chapman Hall / CRC: Chapman Hall / CRC.
- Shalev-Shwartz, Shai and Shai Ben-David (2014). ***Understanding Machine Learning: From Theory to Algorithms***. New York, NY, USA: Cambridge University Press.
- Spearman, Charles (1904). "**" General Intelligence," Objectively**