



Intro into Machine Learning

Why ML works. Overfitting. Model Selection. Figures of Merits. Linear Models. Regularization. Logistic Regression.

Third Machine Learning in High Energy Physics Summer School,
MLHEP 2017, July 17–23

Alexey Artemov^{1,2}

¹ Yandex LLC

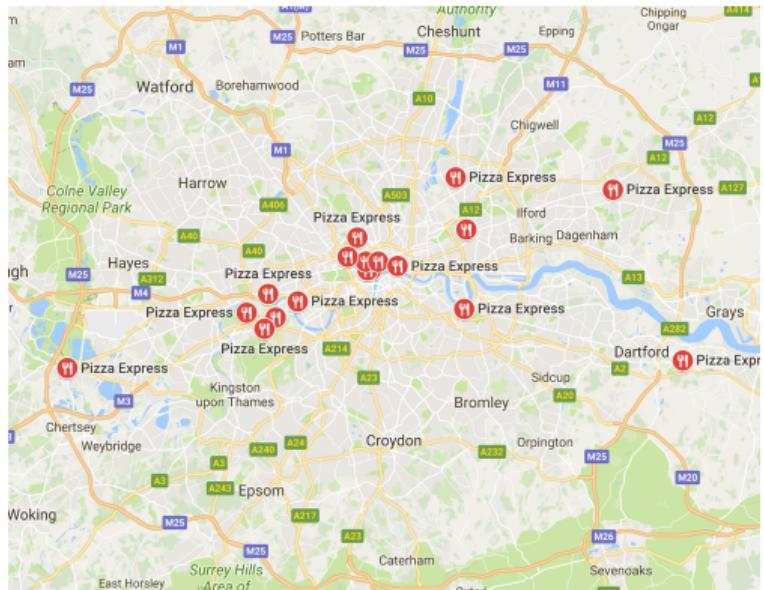
² National Research University Higher School of Economics

Lecture overview

- › A gentle introduction: ML for the real world
- › Linear models: the regression case
- › Linear models: the classification case
- › Figures of Merits: visual evaluation of solution effectiveness
- › Why ML works: a bit of Learning Theory
- › Overfitting: how to fool the linear regression
- › Regularization

Machine learning for the real world

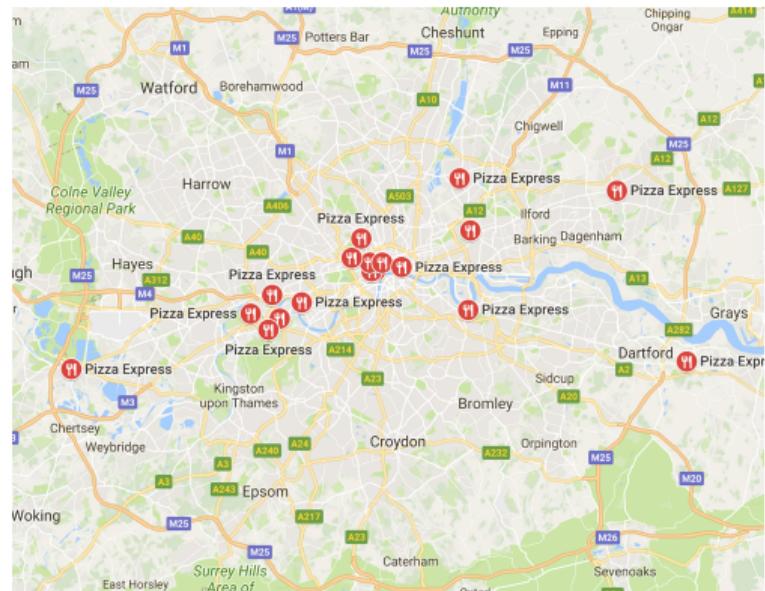
- › You own a pizza restaurant chain
- › Where to open the next restaurant?
- › No revenue laws (of the kind $\vec{F} = m\vec{a}$) exist!
- › Is there any stats on restaurant revenue?



Machine learning for the real world

- › Use stats!
- › Properties and revenues of existing restaurants:

Id	Location	Revenue
1	51.472866,-0.2588277	\$1.1M
2	51.4865675,-0.2210999	\$2.23M
...
20	51.4334641,-0.5146408	\$0.59



Some notation

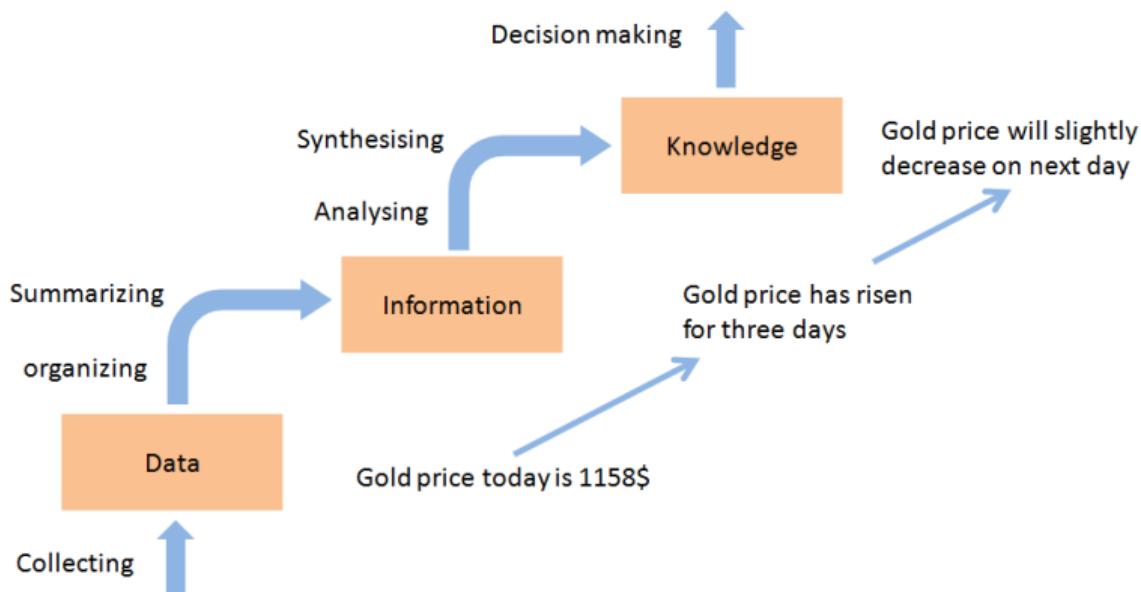
- › Object space $\mathbb{X} = \mathbb{R}^2$: restaurant location (also known as the feature space)
- › Target space $\mathbb{Y} = \mathbb{R}$: restaurant revenue
- › Training set $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$ (known locations and revenues for existing restaurants)
- › **The goal:** estimate $y \in \mathbb{Y}$ for a new $\mathbf{x} \in \mathbb{X}$ (estimate revenue for a proposed location)

Expert knowledge and physical models VS machine learned models

- › Would one want to apply machine learning for ...
 - › ... Newton's mechanics?
 - › ... suggesting music to radio listeners?
 - › ... sorting integers?
 - › ... controlling steel production?
 - › ... sorting strawberries?
- › Criteria for machine learning to be applied in a dependency recovery setting:
 - › No prior knowledge for the dependency exists
 - › The dependency has a complex form too hard for manual examination
 - › A sample from the dependency of sufficiently large size is available

What is machine learning about?

- › Inference of statistical dependencies which give us ability to predict
- › Data is cheap, knowledge is precious



Feature vectors

- › Computers cannot operate objects of the real world directly
- › Need **features** describing the objects ($\mathbf{x} = (\text{lat}, \text{lon})$ for the restaurant)
- › In real applications such as fraud detection, **feature engineering** is the most complex task
- › Many papers on the problem
 - › Machine location (e.g. beside supermarket, latitude, longitude), brand, and age
 - › Customer demographic (e.g. postcode) and behavioral information
 - › Card type (e.g. Visa, supplementary, issued yesterday) and usage
 - › Session/transaction day-of-week, time-of-day, keystrokes (e.g. number of login attempts), transaction type, and amount

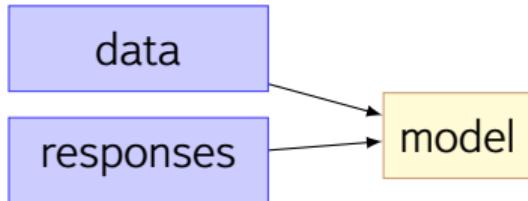


Possible kinds of features: some more notation

- › Binary: $x \in \{0, 1\}$ (are you a member of a terrorist organization?)
- › Real-valued: $x \in \mathbb{R}$ (customer income)
- › Categorical: $x \in S$ where $|S| = n$ and order of elements in S does not matter
(customer postcodes)
- › Ordinal: $x \in S$ where $|S| = n$ and order of elements in S does matter
(old < renovated < new for house price prediction)
- › Structured: images (customer photo in the fraud detection task)

Kinds of machine learning tasks

- › Supervised learning

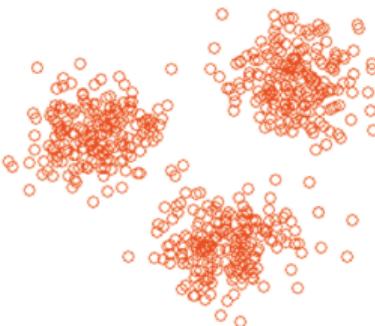
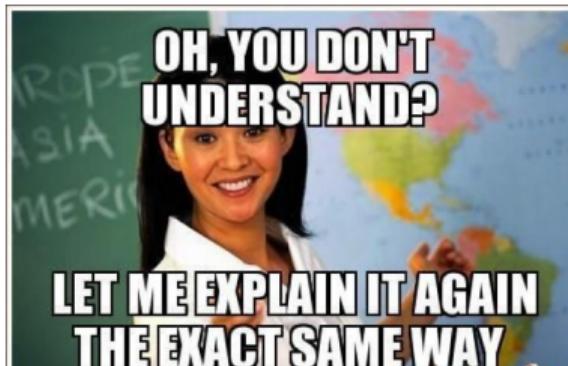


- › Regression, (multi-class/multi-label) classification, semi-supervised learning

- › Unsupervised learning



- › Clustering, density estimation, dimensionality reduction, visualization



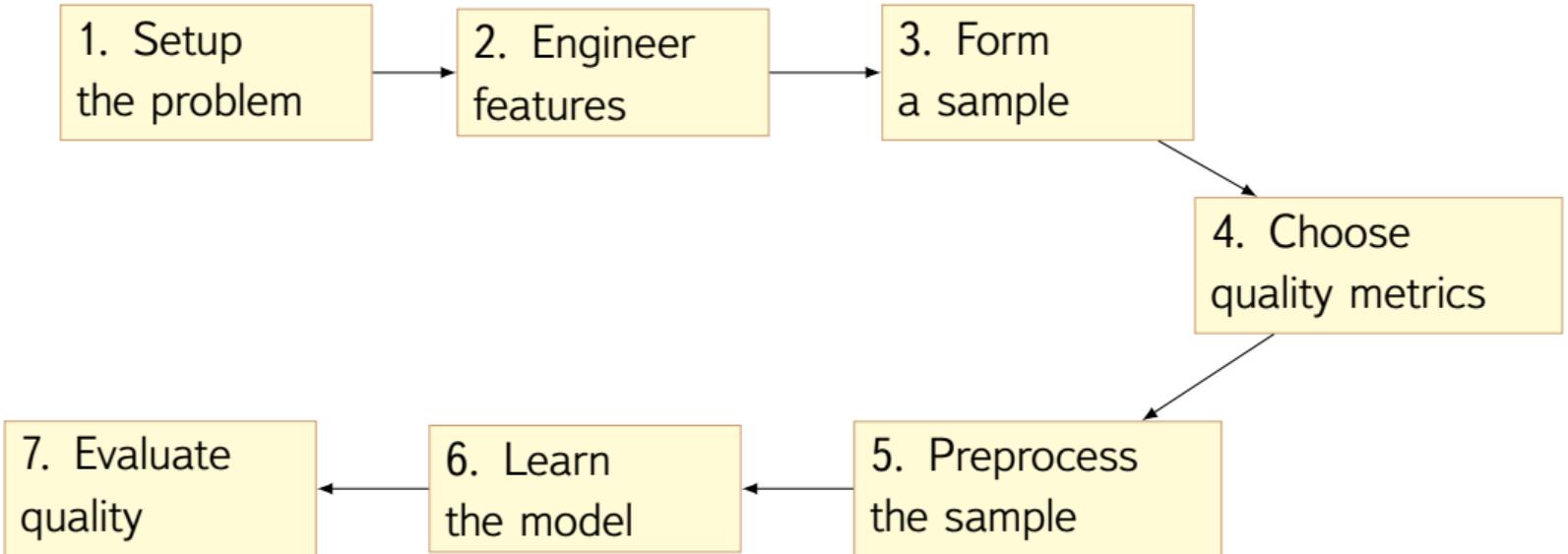
Yet more notation

- › An unknown distribution D generates **instances** $(\mathbf{x}_1, \mathbf{x}_2, \dots)$ independently
- › An unknown function $f : \mathbb{X} \rightarrow \mathbb{Y}$ generates **responses** (y_1, y_2, \dots) for them such that $y_i = f(\mathbf{x}_i), i = 1, 2, \dots$
- › The machine learning problem: choose a plausible **hypothesis** $h : \mathbb{X} \rightarrow \mathbb{Y}$ from the **hypothesis space** \mathbb{H}
- › The error of a hypothesis h is the deviation from the true f measured by the **loss function** (an example for regression):

$$Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} (f(\mathbf{x}_i) - h(\mathbf{x}_i))^2$$

- › **Learning:** the search for the optimal hypothesis $h \in \mathbb{H}$ w.r.t. the fixed loss function

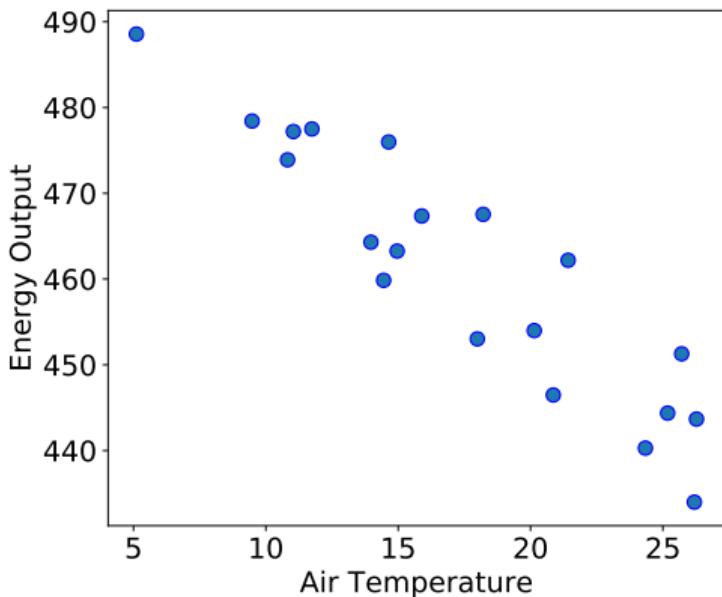
The machine learning pipeline



Linear models for regression

Univariate linear regression

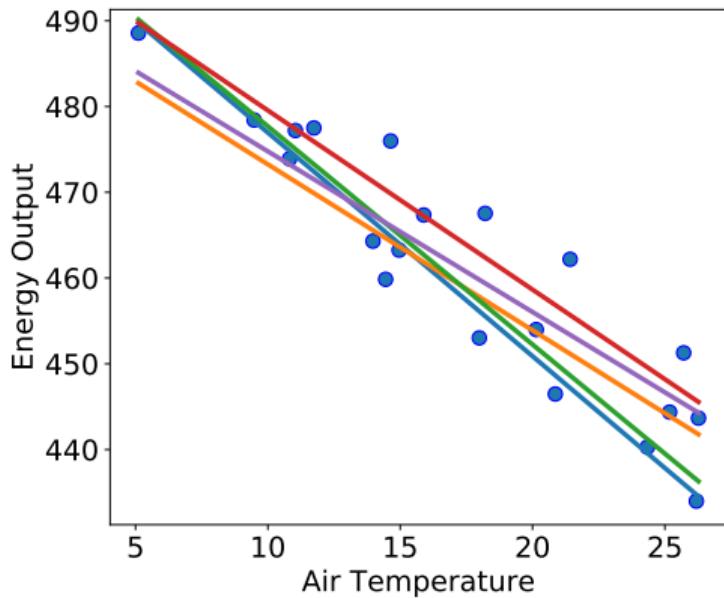
- › A single feature (**regressor**) x :
Air Temperature
- › A single **dependent variable** y :
Energy Output
- › Training set $X^\ell = \{(x_i, y_i)\}_{i=1}^{20}$
- › The regression model:
$$y_i = h(x_i; \mathbf{w}) + \varepsilon_i$$
- › Linear model: $y_i = w_1 x_i + w_0 + \varepsilon_i$
- › **The goal:** given X^ℓ , find $\mathbf{w} = (w_1, w_0)$



Univariate linear regression

- › Which fit to choose?
- › With the linear model being fixed, depends on the data and the loss function!
- › Mean square (L2) loss (MSE):

$$Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} (y_i - h(x_i))^2$$



Some other evaluation metrics for regression

- › Mean square (L2) loss (MSE): $\text{MSE}(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} (y_i - h(x_i))^2$
- › Root MSE: $\text{RMSE}(h, X^\ell) = \sqrt{\frac{1}{\ell} \sum_{i=1}^{\ell} (y_i - h(x_i))^2}$
- › Coefficient of determination: $R^2(h, X^\ell) = 1 - \frac{\sum_{i=1}^{\ell} (y_i - h(x_i))^2}{\sum_{i=1}^{\ell} (y_i - \mu_y)^2}$
with $\mu_y = \frac{1}{\ell} \sum_{i=1}^{\ell} y_i$
- › Mean absolute error: $\text{MAE}(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} |y_i - h(x_i)|$

Univariate linear regression

- With the loss fixed, the linear problem reduces to optimization:

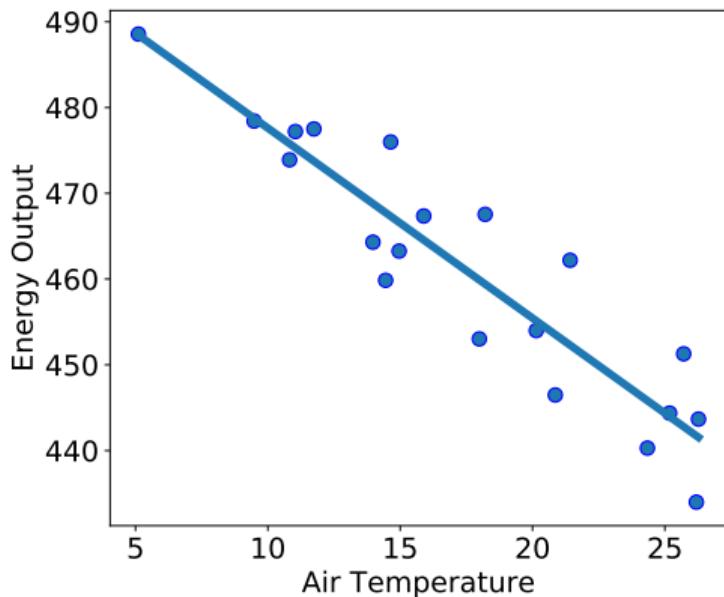
$$\frac{1}{\ell} \sum_{i=1}^{\ell} (y_i - w_1 x_i - w_0)^2 \rightarrow \min_{(w_0, w_1) \in \mathbb{R}^2},$$

to which an analytical solution is available

$$\hat{w}_1 = \frac{\sum_{i=1}^{\ell} (x_i - \mu_x)(y_i - \mu_y)}{\sum_{i=1}^{\ell} (x_i - \mu_x)^2},$$

$$\hat{w}_0 = \mu_y - \hat{w}_1 \mu_x$$

$$\text{with } \mu_x = \frac{1}{\ell} \sum_{i=1}^{\ell} x_i, \quad \mu_y = \frac{1}{\ell} \sum_{i=1}^{\ell} y_i$$



Multivariate linear regression

- › Multiple features (regressors) $\mathbf{x}_i = (x_{1i}, \dots, x_{di})$ available for each y_i
- › The model:

$$y_1 = w_1 x_{11} + \dots + w_d x_{d1} + \varepsilon_1,$$

$$y_2 = w_1 x_{12} + \dots + w_d x_{d2} + \varepsilon_2,$$

...

$$y_\ell = w_1 x_{1\ell} + \dots + w_d x_{d\ell} + \varepsilon_\ell,$$

is often written in matrix-vector form as

$$\begin{bmatrix} y_1 \\ \vdots \\ y_\ell \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \dots & x_{d1} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1\ell} & x_{2\ell} & \dots & x_{d\ell} \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_\ell \end{bmatrix} \quad \longleftrightarrow \quad \mathbf{y} = \mathbf{X}\mathbf{w} + \boldsymbol{\varepsilon}$$

Multivariate linear regression: the solution

- › The problem: minimize MSE

$$Q(h, X^l) = \sum_{i=1}^{\ell} \left(y_i - \sum_{k=1}^d w_k x_{ki} \right)^2 \equiv \| \mathbf{y} - \mathbf{Xw} \|^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › Solve analytically via computing the gradient

$$\nabla_{\mathbf{w}} \| \mathbf{y} - \mathbf{Xw} \|^2 = 2(\mathbf{y} - \mathbf{Xw})\mathbf{X} = 0$$

- › The solution

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

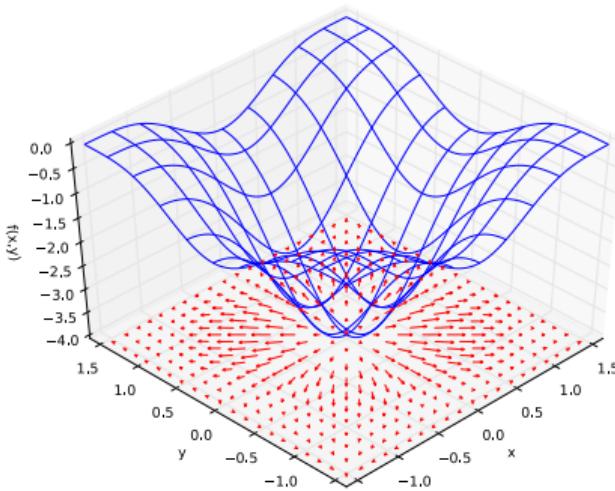
A quick intro into the Numerical Optimization

- Consider the optimization problem in \mathbb{R}^d

$$f(\mathbf{x}) \rightarrow \min_{\mathbf{x} \in \mathbb{R}^d} \quad (\text{such as } \sum_{i=1}^{\ell} \left(y_i - \sum_{k=1}^d w_k x_{ki} \right)^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d})$$

- In general, solved using the numerical methods such as the gradient descent
- Gradients: directions in \mathbb{R}^d pointing towards steepest function increase

$$\nabla_{\mathbf{x}} f(\mathbf{x}) \equiv \left(\frac{\partial f(\mathbf{x})}{\partial x_1}, \dots, \frac{\partial f(\mathbf{x})}{\partial x_d} \right)$$



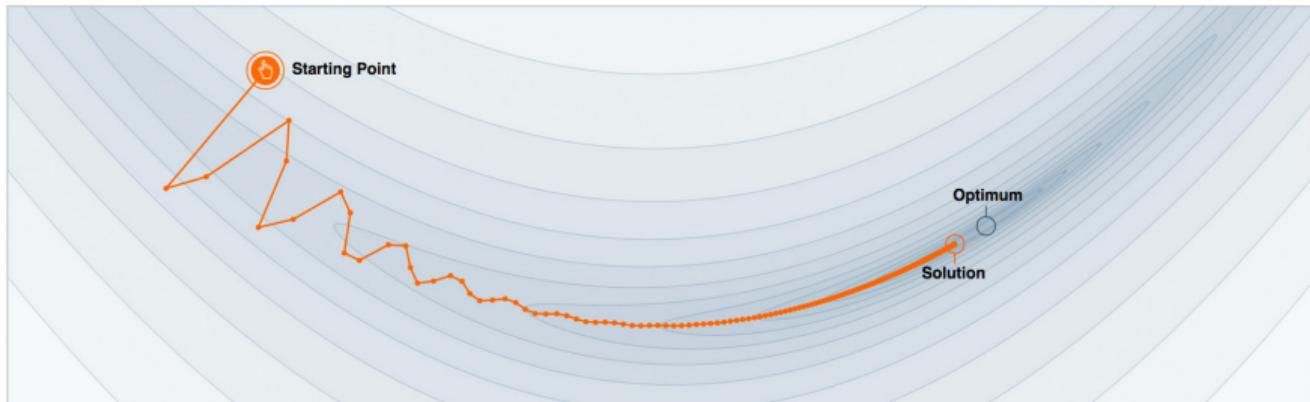
The gradient descent algorithm

- › The gradient descent procedure iterates from $\mathbf{x}^{(0)}$ as

$$\mathbf{x}^{(k)} \leftarrow \mathbf{x}^{(k-1)} - \alpha_k \nabla_{\mathbf{x}} f(\mathbf{x}^{(k-1)})$$

with α_k controlling the k th step size

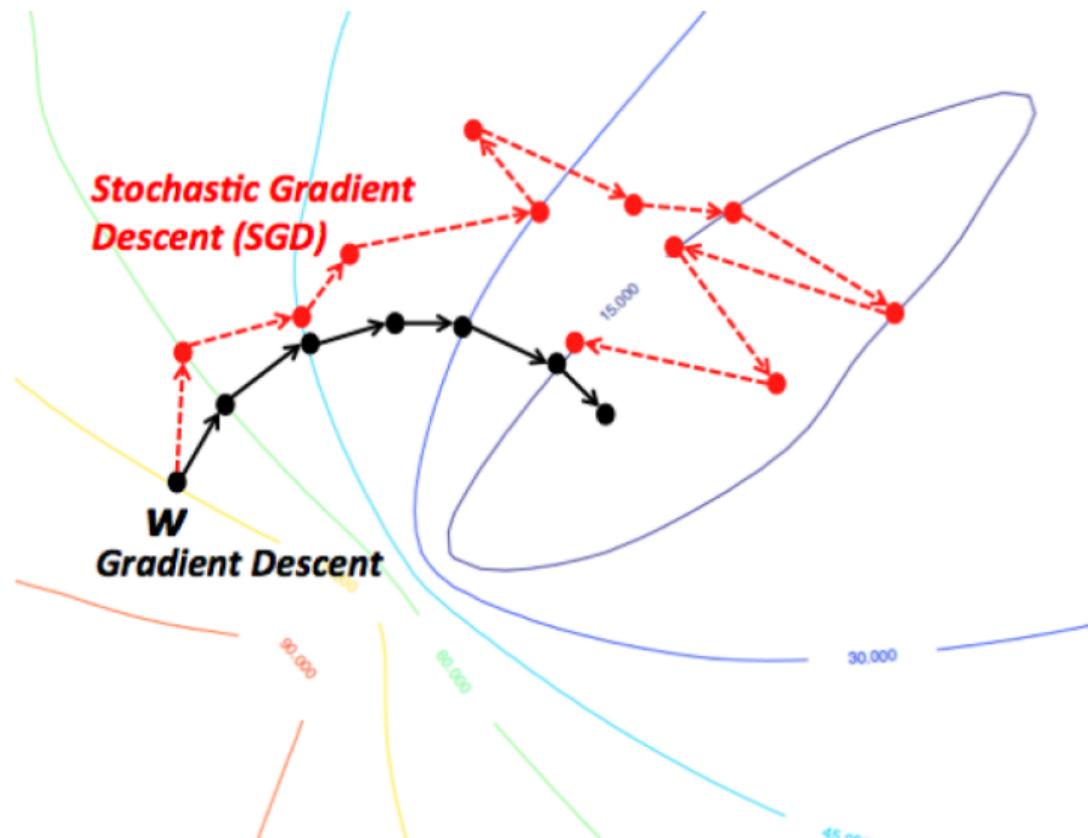
- › For smooth convex functions with a single minimum \mathbf{x}^* , k steps of gradient descent achieve accuracy $f(\mathbf{x}^{(k)}) - f(\mathbf{x}^*) = \mathcal{O}(1/k)$



The stochastic gradient descent algorithm

- › Machine learning: many additive targets $f(\mathbf{w}) = \sum_{i=1}^{\ell} f_i(\mathbf{w})$,
computationally inefficient for large ℓ
- › Use subsamples for gradient estimation: the Stochastic Gradient Descent (SGD)
 1. Pick $i_k \in \{1, \dots, \ell\}$ at random;
 2. Compute $\mathbf{w}^{(k)} \leftarrow \mathbf{w}^{(k-1)} - \alpha_k \nabla_{\mathbf{w}} f_{i_k}(\mathbf{w}^{(k-1)})$
- › For smooth convex functions with a single minimum \mathbf{x}^* , k steps of SGD achieve accuracy $f(\mathbf{w}^{(k)}) - f(\mathbf{w}^*) = \mathcal{O}(1/\sqrt{k})$
- › Batch, variance reduction, momentum hacks available to improve the convergence rate to $\mathcal{O}(1/k)$

Gradient descent vs stochastic gradient descent



Multivariate linear regression: the numerical solution

- › Initialize with some $\mathbf{w}^{(0)}$
- › Gradient in i_k th object is

$$\nabla_{\mathbf{x}} f_{i_k}(\mathbf{w}) = 2(y_{i_k} - \mathbf{x}_{i_k}^\top \mathbf{w}) \mathbf{x}_{i_k} \quad (\in \mathbb{R}^d)$$

- › Compute updates using SGD:

$$\mathbf{w}^{(k)} \leftarrow \mathbf{w}^{(k-1)} - \alpha_k \nabla_{\mathbf{w}} f_{i_k}(\mathbf{w}^{(k-1)})$$

Linear models for classification

Linear models for classification

- › An unknown distribution D generates instances $(\mathbf{x}_1, \mathbf{x}_2, \dots)$
- › An unknown function $f : \mathbb{X} \rightarrow \mathbb{Y}$ generates labels (y_1, y_2, \dots) for them such that $y_i = f(\mathbf{x}_i)$, and $y_i \in \{-1, +1\}$
- › The classification problem: choose a plausible hypothesis (**classifier**) $h : \mathbb{X} \rightarrow \mathbb{Y}$ from the **hypothesis space** \mathbb{H}
- › The error of the classifier h is the probability (over D) that it will fail

$$Q(h, D) = \Pr_{\mathbf{x} \sim D}[f(\mathbf{x}) \neq h(\mathbf{x})]$$

usually estimated by the **accuracy metric**

$$Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} [f(\mathbf{x}_i) \neq h(\mathbf{x}_i)]$$

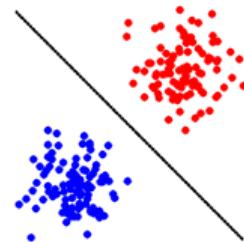
Linear models for classification

- › Linear model: $h(\mathbf{x}) = \text{sign}\left(\sum_{i=1}^d w_i x_i + w_0\right) = \text{sign}(\mathbf{w}^\top \mathbf{x} + w_0)$
- › The learning problem is discrete over $\mathbf{w} \in \mathbb{R}^d$:

$$Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} [\text{sign}(\mathbf{w}^\top \mathbf{x}_i) \neq h(\mathbf{x}_i)] \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

(cannot optimize using gradient descent)

- › The solution: optimize a differentiable upper bound for $Q(h, X^\ell)$!
- › $Q(h, X^\ell)$ can be written using $Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} L(M_i)$
 - where $L(M_i) = [M_i < 0] \equiv [y_i \mathbf{w}^\top \mathbf{x}_i < 0]$
- › Upper-bounding $L(M)$ yields upper bounds for $Q(h, X^\ell)$

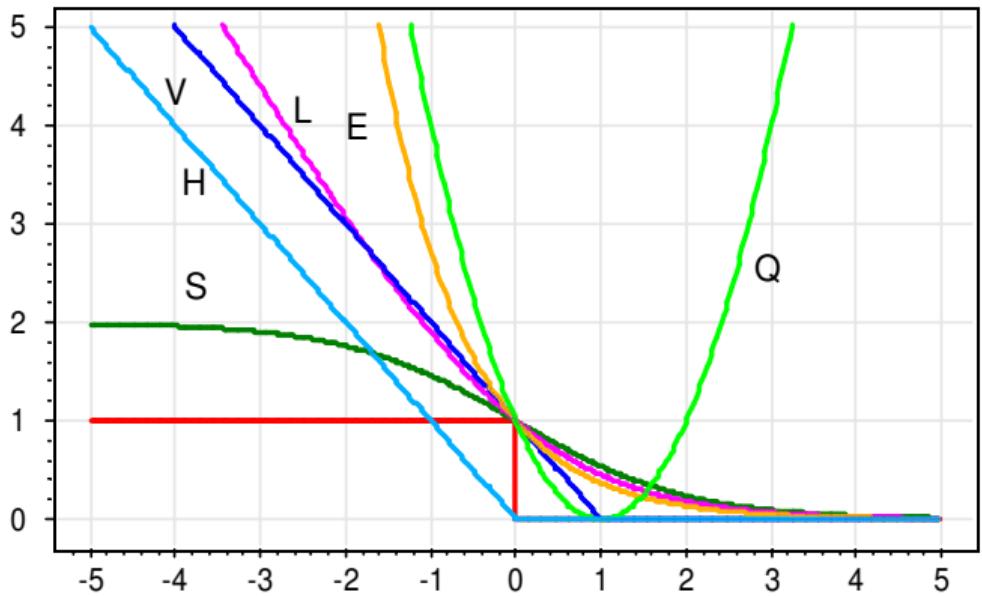


Linear models for classification: upper bounds

Multiple approximations to accuracy loss

- › $L_L(M) = \log(1 + e^{-M})$
- › $L_H(M) = \max(0, 1 - M)$
- › $L_P(M) = \max(0, -M)$
- › $L_E(M) = e^{-M}$
- › $L_S(M) = 2/(1 + e^M)$

give rise to various learning algorithms



The logistic regression model

- › Training set $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$ where $y_i \in \{-1, +1\}$
- › We seek an algorithm h such that $h(\mathbf{x}) = P(y = +1 | \mathbf{x})$
- › A probability that an instance (\mathbf{x}_i, y_i) is encountered in X^ℓ

$$h(\mathbf{x}_i)^{[y_i=+1]} + (1 - h(\mathbf{x}_i))^{[y_i=-1]}$$

- › Entire X^ℓ likelihood:

$$L(X^\ell) = \prod_{i=1}^{\ell} h(\mathbf{x}_i)^{[y_i=+1]} + (1 - h(\mathbf{x}_i))^{[y_i=-1]}$$

is often written via **log-likelihood** (of which the negative is **log-loss**)

$$\log L(X^\ell) = \sum_{i=1}^{\ell} [y_i = +1] \log h(\mathbf{x}_i) + [y_i = -1] \log(1 - h(\mathbf{x}_i))$$

The logistic regression model

- › The choice of h : sigmoid function

$$h(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x})$$

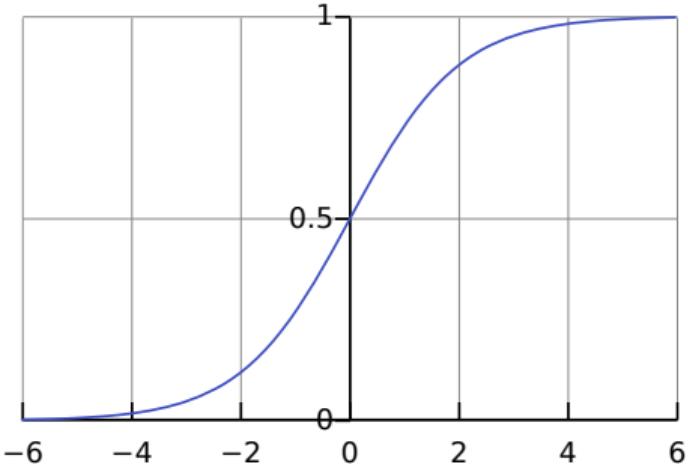
where $\sigma(x) \in [0, 1]$

- › Typical choice: the logistic function

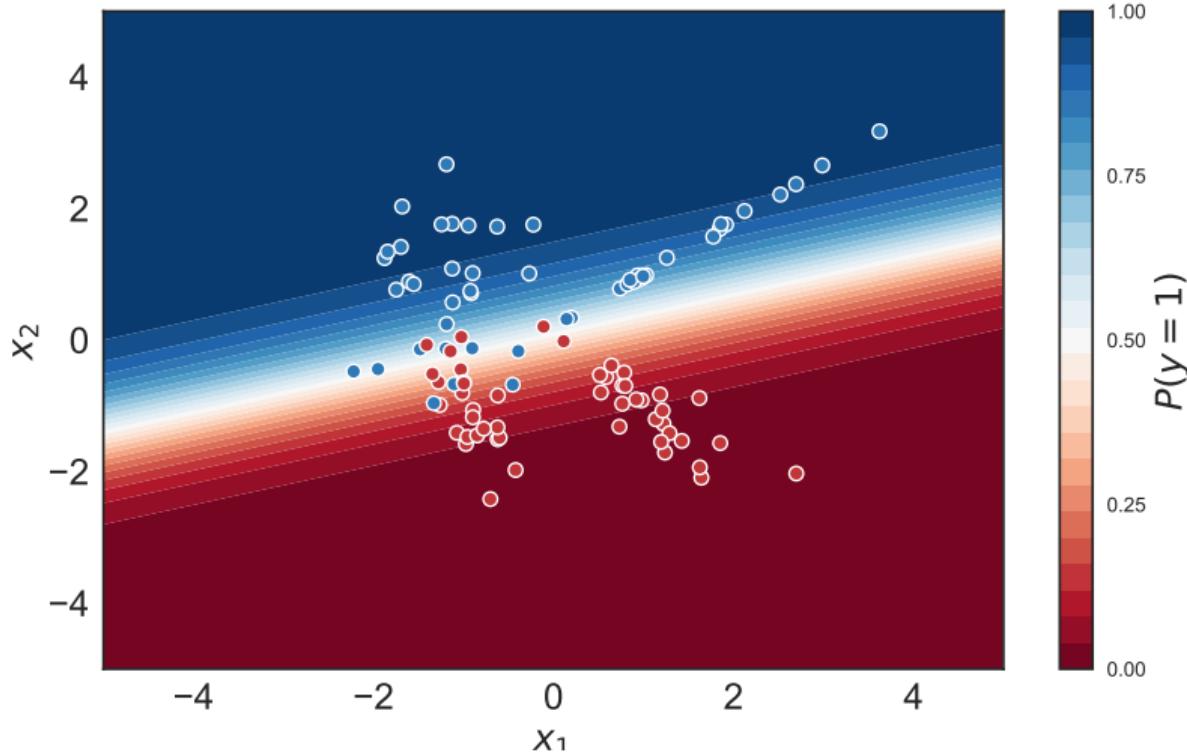
$$\sigma(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})}$$

- › Plugging the logistic function into the loss yields

$$\sum_{i=1}^{\ell} (1 + \exp(\mathbf{w}^\top \mathbf{x})) \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$



The logistic regression model



Figures of Merits

Classification quality evaluation: accuracy

- › Given a labeled sample $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$, $y_i \in \{-1, +1\}$, and some candidate h , how well does h perform on X^ℓ ?
- › Let $a(x) = [h(x) > t]$
- › Obvious choice: accuracy

$$\text{accuracy}(a, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} [a(\mathbf{x}_i) = y_i]$$

- › Bad for **imbalanced data**: for $\ell = 1000$,
 $n_- = \sum_{i=1}^{\ell} [y = -1] = 50$, $n_+ = \sum_{i=1}^{\ell} [y = +1] = 950$,
a trivial rule $h(\mathbf{x}) = +1$ would yield $\text{accuracy}(a, X^\ell) = 0.95$

Classification quality: confusion matrix

	$y = 1$	$y = -1$
$a(x) = 1$	True Positive (TP)	False Positive (FP)
$a(x) = -1$	False negative (FN)	True Negative (TN)

- › More informative criteria:

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- › While accuracy can be expressed, too

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$

Classification quality: operating curves

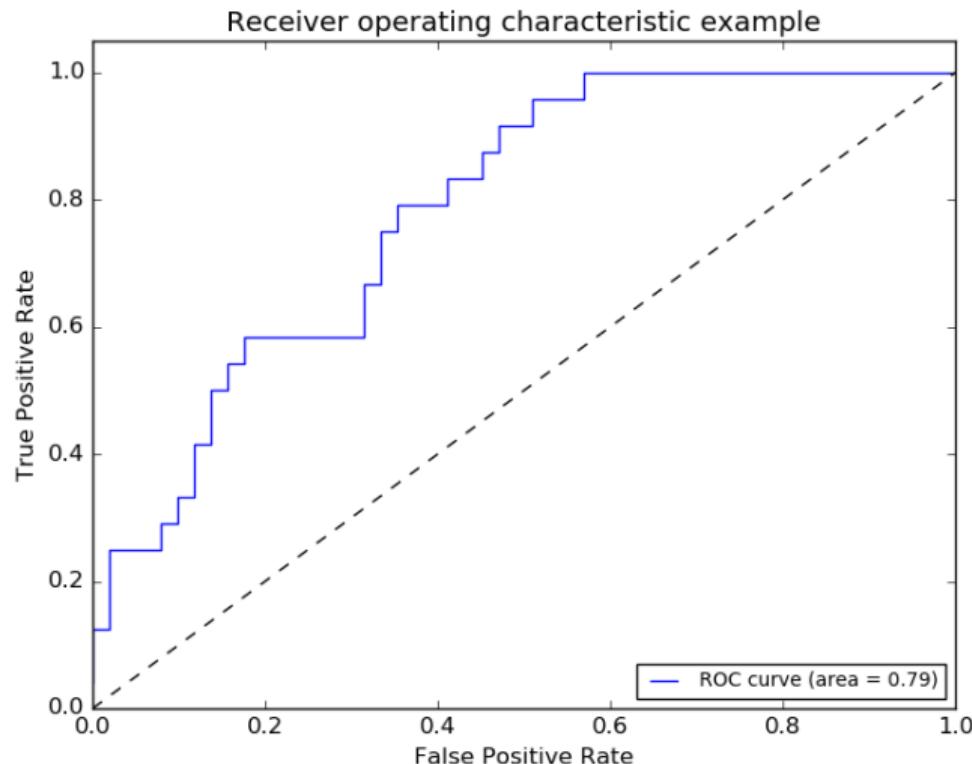
- › Often $h(\mathbf{x})$ is more valuable than its thresholded version $a(x) = [h(x) > t]$
- › Consider two-dimensional space with coordinates

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \quad \text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

corresponding to various choices of the threshold t

- › The plot TPR(FPR) is called the **receiver operating characteristic** (or ROC) curve
- › Area under curve (ROC-AUC) reflects classification quality

Receiver operating characteristic curve

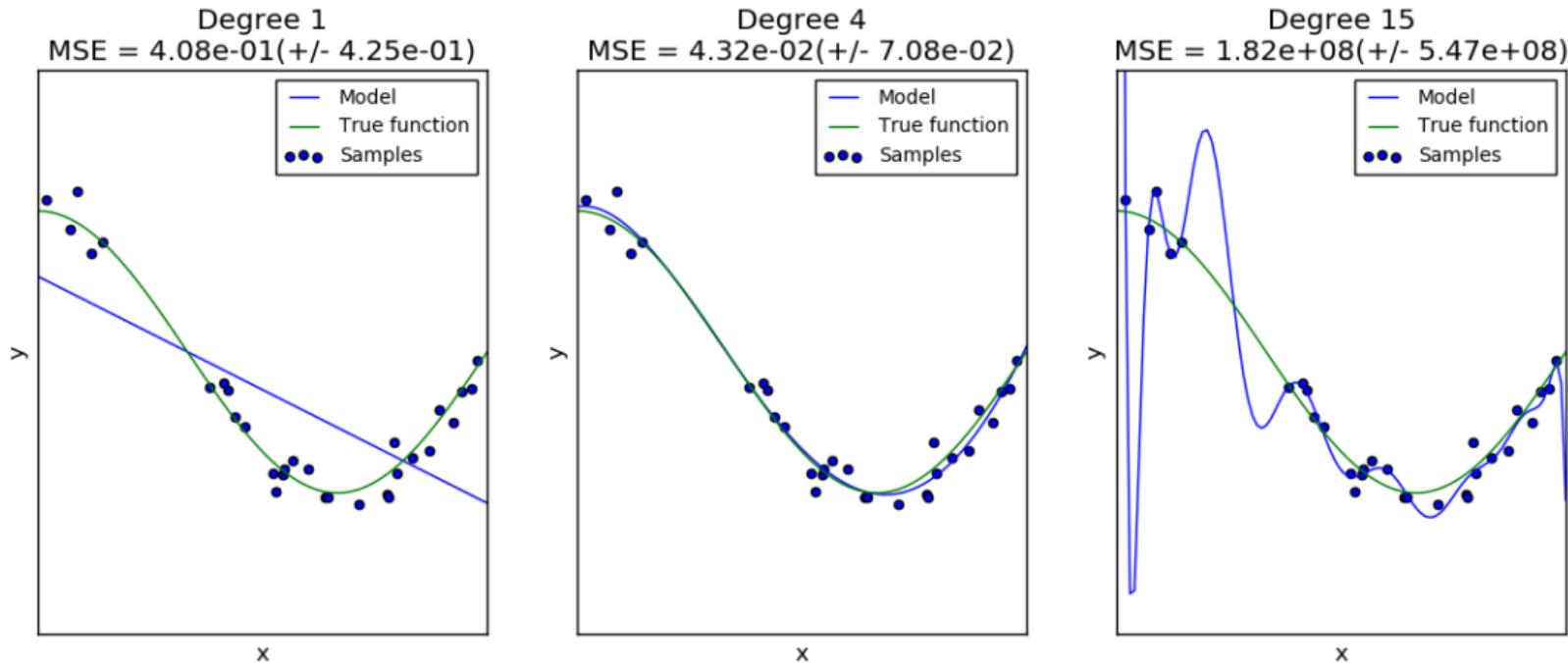


Overfitting: how to fool
the linear regression

Generalization and overfitting

- › Training set memorization: for seen $(\mathbf{x}, y) \in X^\ell$, $h(\mathbf{x}) = y$
- › Generalization: equally good performance on both new and seen instances
- › How to assess model's generalization ability?
- › Consider an example:
 - › $y = \cos(1.5\pi x) + \mathcal{N}(0, 0.01)$, $x \sim \text{Uniform}[0, 1]$
 - › Features: $\{x\}$, $\{x, x^2, x^3, x^4\}$, $\{x, \dots, x^{15}\}$
- › How well do the regression models perform?

Polynomial fits of different degrees



Assessing generalization ability: cross-validation

- › Split training set into subsets of equal size $X^\ell = X_1^\ell \cup \dots \cup X_K^\ell$
- › Train K models h_1, \dots, h_K where each model h_k is trained on all subsets but X_k^ℓ
- › Assess quality using $\text{CV} = \frac{1}{K} \sum_{k=1}^K Q(h_k, X_k^\ell)$ (K -fold)
- › Leave-one-out cross-validation: $X_k^\ell = \{(\mathbf{x}_k, y_k)\}$

Regularization

Ad-hoc regularization: motivation

- › Consider the multivariate linear regression problem with $\mathbf{X} \in \mathbb{R}^{d \times d}$

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › Analytic solution involves computing the product $\mathbf{R} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$
- › If $\mathbf{X} = \text{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_1 > \lambda_2 > \dots > \lambda_d \rightarrow 0$
(meaning we're in eigenbasis of \mathbf{X}) then

$$\begin{aligned}\mathbf{R} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top = \\ &= \left(\text{diag}(\lambda_1, \dots, \lambda_d) \text{diag}(\lambda_1, \dots, \lambda_d) \right)^{-1} \text{diag}(\lambda_1, \dots, \lambda_d) = \\ &= \text{diag} \left(\frac{1}{\lambda_1}, \dots, \frac{1}{\lambda_d} \right), \quad \text{leading to huge diagonal values in } \mathbf{R}\end{aligned}$$

Ad-hoc regularization: L2

- › Regularization: replace **fit** with **fit + penalty** as in

$$Q(\mathbf{w}) \rightarrow Q_\alpha(\mathbf{w}) = Q(\mathbf{w}) + \alpha R(\mathbf{w})$$

- › $R(\mathbf{w})$ is called the regularizer, $\alpha > 0$ – the regularization constant
- › Regularized multivariate linear regression problem

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \alpha\|\mathbf{w}\|_2^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › Regularized analytic solution available

$$\mathbf{w}^* = (\mathbf{X}^\top \mathbf{X} + \alpha \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$$

Why L2 regularization works

- › Analytic solution: compute the regularized operator

$$\mathbf{R} = (\mathbf{X}^\top \mathbf{X} + \alpha \mathbf{I})^{-1} \mathbf{X}^\top$$

- › If $\mathbf{X} = \text{diag}(\lambda_1, \dots, \lambda_d)$ with $\lambda_1 > \lambda_2 > \dots > \lambda_d \rightarrow 0$ (meaning we're in eigenbasis of \mathbf{X}) then

$$\begin{aligned}\mathbf{R} &= (\mathbf{X}^\top \mathbf{X} + \alpha \mathbf{I})^{-1} \mathbf{X}^\top = \\ &= (\text{diag}(\lambda_1, \dots, \lambda_d) \text{diag}(\lambda_1, \dots, \lambda_d) + \text{diag}(\alpha, \dots, \alpha))^{-1} \text{diag}(\lambda_1, \dots, \lambda_d) = \\ &= \text{diag}\left(\frac{\lambda_1}{\lambda_1^2 + \alpha}, \dots, \frac{\lambda_d}{\lambda_d^2 + \alpha}\right),\end{aligned}$$

smoothing diagonal values in \mathbf{R}

More regularizers!

- › L2 regularized multivariate linear regression problem

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \alpha\|\mathbf{w}\|^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › L1 regularized regression (LASSO)

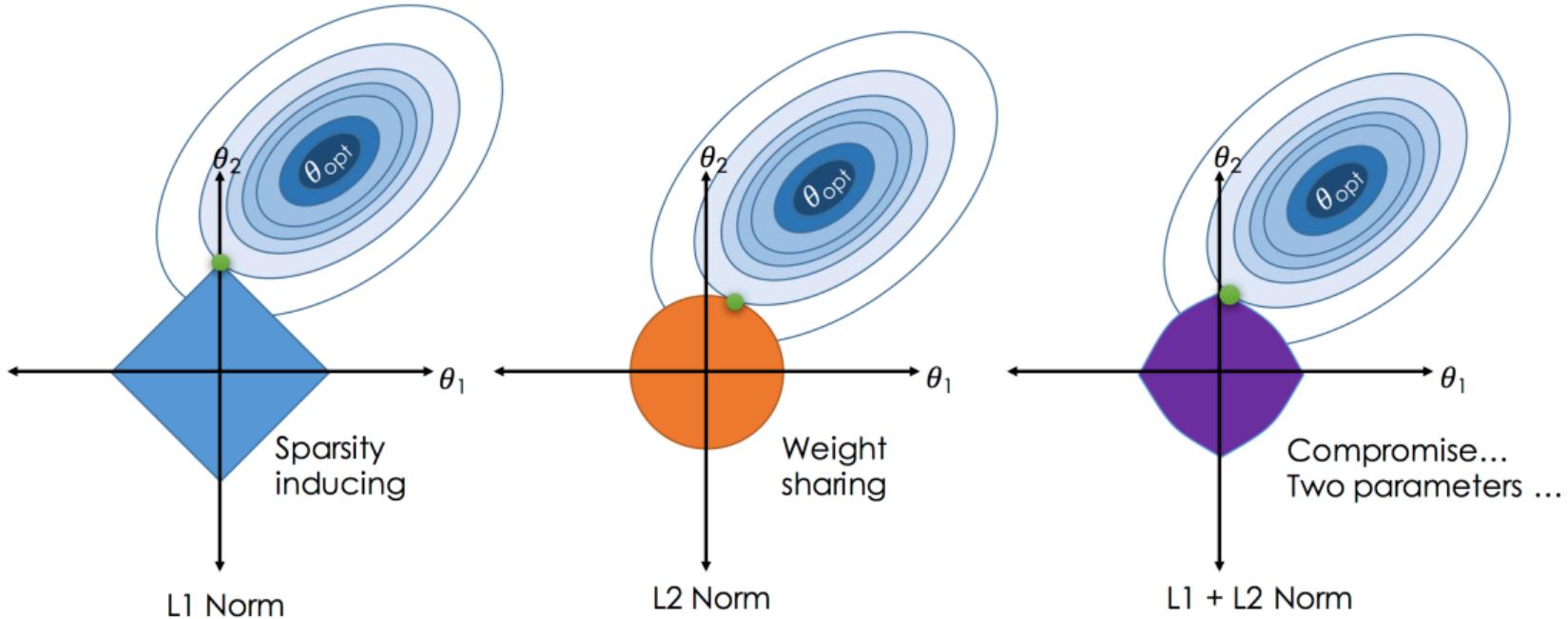
$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \alpha\|\mathbf{w}\|_1 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › L1/L2 regularized regression (Elastic Net)

$$\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \alpha_1\|\mathbf{w}\|_1 + \alpha_2\|\mathbf{w}\|_2^2 \rightarrow \min_{\mathbf{w} \in \mathbb{R}^d}$$

- › Convex $Q(\mathbf{w})$: **unconstrained** optimization $Q(\mathbf{w}) + \alpha\|\mathbf{w}\|_1$
is equivalent to **constrained** problem $Q(\mathbf{w})$ s.t. $\|\mathbf{w}\|_1 \leq C$

Geometric interpretation of regularizers



Picture credit: http://www.ds100.org/sp17/assets/notebooks/linear_regression/Regularization.html

Another interpretation of regularizers



Figure: Large parameter space



Figure: Regularized models

Why Machine Learning works:
a bit of Learning Theory

Empirical Risk Minimization framework

- › Given a labeled sample $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$ and some candidate h
- › Define the **empirical error** of h as $Q(h, X^\ell) = \frac{1}{\ell} \sum_{i=1}^{\ell} [f(\mathbf{x}_i) \neq h(\mathbf{x}_i)]$
(the proportion of sample points on which h errs)
- › $\text{ERM}(X^\ell)$ — find h the minimizes $Q(h, X^\ell)$
- › **No learning is possible without applying prior knowledge**
- › A hypothesis class \mathbb{H} is a set of hypotheses
- › Re-define the ERM rule by searching only inside such a prescribed \mathbb{H}
- › $\text{ERM}_{\mathbb{H}}(X^\ell)$ picks a classifier $h \in \mathbb{H}$ that minimizes the empirical error over members of \mathbb{H}

Guarantees for ERM

Theorem (Guaranteed success for $\text{ERM}_{\mathbb{H}}$)

- › Let \mathbb{H} be a finite class.
- › Let the unknown labeling rule, f , be a member of \mathbb{H} .
- › Then
 - › for every $\varepsilon, \delta > 0$, if $m > (\log(|\mathbb{H}|) + \log(1/\delta))/\varepsilon$,
 - › with probability $> 1 - \delta$ (over the choice of X^ℓ),
 - › any $\text{ERM}_{\mathbb{H}}$ hypothesis has error below ε .

Bonus track

Structural Risk Minimization framework

- › Balancing the model's complexity against its success at fitting the training data
- › Given \mathbb{H} , a decompositon of \mathbb{H} to 'good' \mathbb{H}_n 's and a weight functon w ,
- › On a labeled training sample S of size m ,
- › Find $h \in \mathbb{H}$ that minimizes

$$Q(h, X^\ell) + \min\{n : h \in \mathbb{H}_n\} \varepsilon_n(m, w(n)\delta)$$

where $\varepsilon_n(m, \delta) = \min\{\varepsilon : m_{\mathbb{H}_n}(\varepsilon, \delta) < m\}$
(namely, the minimal error that an m -size sample can guarantee)

Structural Risk Minimization framework

- › SRM has many applications, usually referred to as “ERM with regularization”
- › Adding to the empirical error a “penalty” on complex (or otherwise, undesirable) h ’s.
- › Examples include:
 1. Norm of a linear classifier
 2. Description length
 3. Small margins
 4. Low prior likelihood

Linear discriminant analysis [R. Fisher, 1936]

- › Training set $X^\ell = \{(\mathbf{x}_i, y_i)\}_{i=1}^\ell$ where $y_i \in \{-1, +1\}$

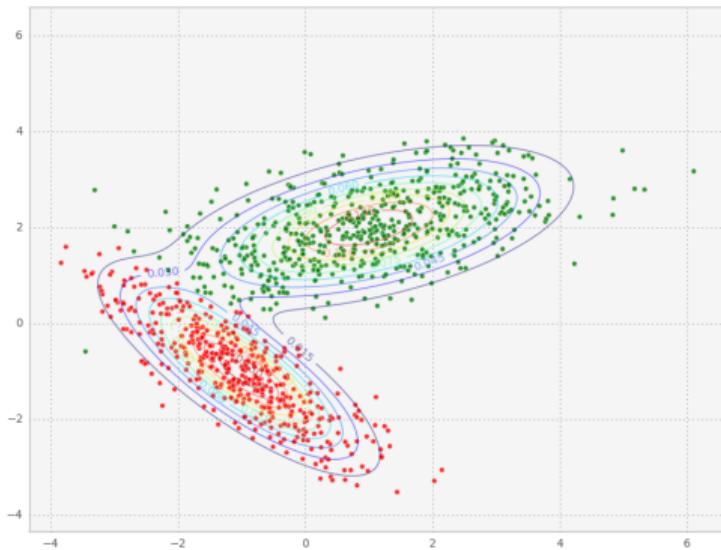
- › The model:

$$P(\mathbf{x}_i | y_i = -1) = \mathcal{N}(\boldsymbol{\mu}_-, \boldsymbol{\Sigma}_-),$$

$$P(\mathbf{x}_i | y_i = +1) = \mathcal{N}(\boldsymbol{\mu}_+, \boldsymbol{\Sigma}_+)$$

- › Minimize error probability via Neyman-Pearson lemma

$$\frac{P(\mathbf{x} | y_i = +1)}{P(\mathbf{x} | y_i = -1)} > t$$



Linear discriminant analysis [R. Fisher, 1936]

- › Log-likelihood ratio

$$\log P(\mathbf{x}|y_i = +1) - \log P(\mathbf{x}|y_i = -1) > T$$

- › Recall that $P(\mathbf{x}|y_i = \pm 1) = \frac{1}{\sqrt{2\pi|\Sigma_{\pm}|}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu}_{\pm})^T \boldsymbol{\Sigma}_{\pm}^{-1} (\mathbf{x} - \boldsymbol{\mu}_{\pm})\right\}$

- › When $\Sigma_+ = \Sigma_- = \Sigma$, the optimal solution is

$$\mathbf{x}^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-) > \frac{1}{2}(T - (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{\mu}_+ - \boldsymbol{\mu}_-))$$

which has the form $\mathbf{w}^T \mathbf{x} > c$

- › Homoscedastic case: linear discriminant hyperplane

Linear discriminant analysis

