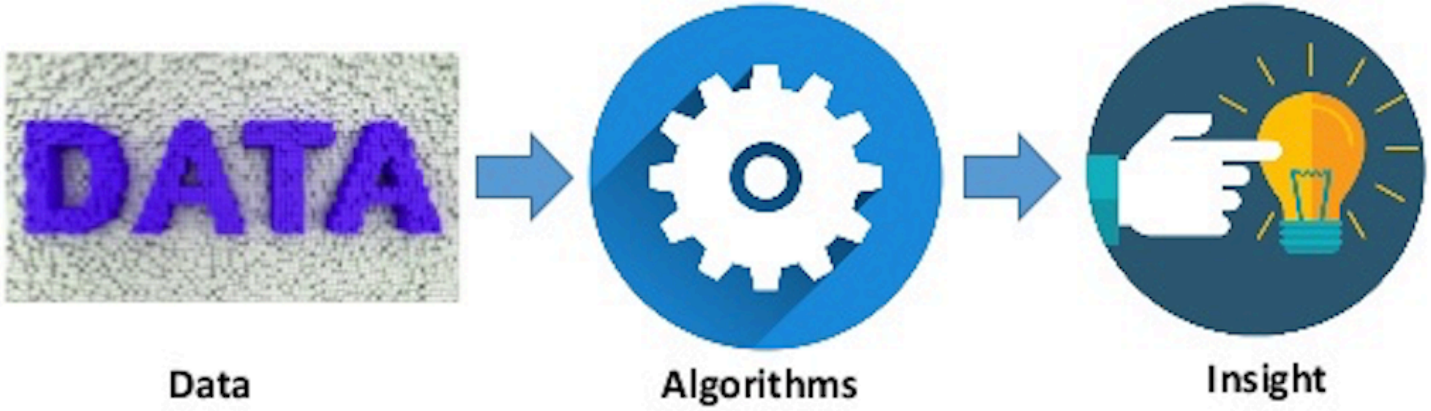
Lecture 2: Linear Regression

Recap: What is Machine Learning?



Recap: What data?

- Attributes:

	Age at delivery	Weight prior to pregnancy (pounds)	Smoker	Doctor visits during 1st trimester	Race	Birth Weight (grams)
Patient 1	29	140	Yes	2	Caucasian	2977
Patient 2	32	132	No	4	Caucasian	3080
Patient 3	36	175	No	0	African-Am	3600
*	*	*	*	*	*	*
*	*	*	*	*	*	*
Patient 189	30	95	Yes	2	Asian	3147

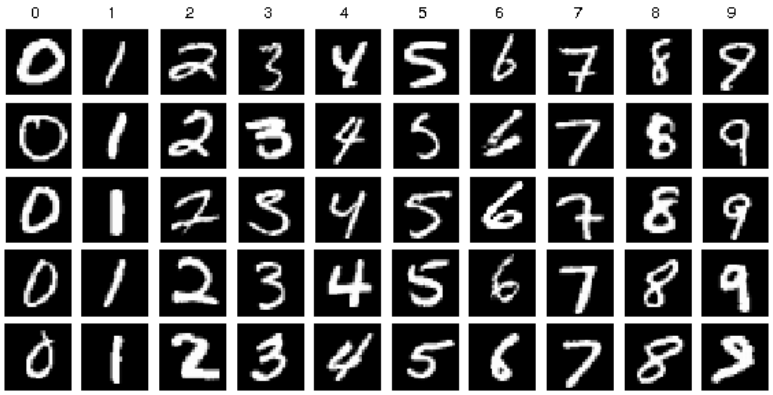
Image from Lumen Learning

- Text:

	Column1	Column2
6	A very, very, very slow-moving, aimless movie about a distressed, drifting young man.	0
8	Not sure who was more lost - the flat characters or the audience, nearly half of whom walked out.	0
10	Attempting artiness with black & white and clever camera angles, the movie disappointed - became e	0
11	Very little music or anything to speak of.	0
13	The best scene in the movie was when Gerardo is trying to find a song that keeps running through his	1
15	The rest of the movie lacks art, charm, meaning... If it's about emptiness, it works I guess because it's	0
16	Wasted two hours.	0
18	Saw the movie today and thought it was a good effort, good messages for kids.	1
20	A bit predictable.	0
22	Loved the casting of Jimmy Buffet as the science teacher.	1
23	And those baby owls were adorable.	1
25	The movie showed a lot of Florida at it's best, made it look very appealing.	1
26	The Songs Were The Best And The Muppets Were So Hilarious.	1

Image from Integrated Knowledge Solutions

- Images



- Speech/sound:

"without the dataset
the article is useless"

Recap: What insight?

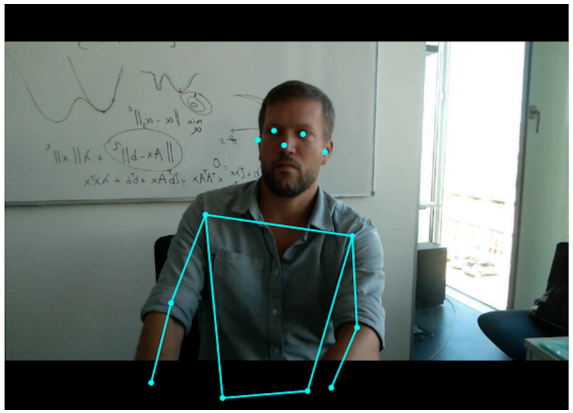
Category

"cat"

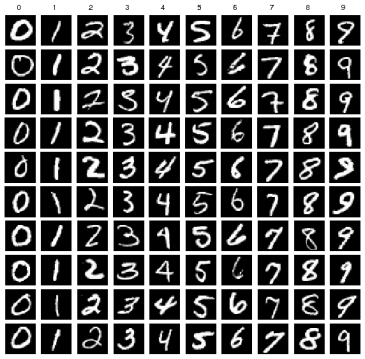
Description

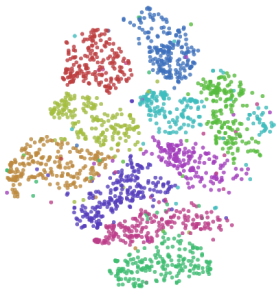
"A young man riding a skateboard"

Vector-valued quantity



Different representation





Recap: What (classes of) algorithms?

- Supervised learning
  - Relies on supervised data
  - The annotations typically correspond to the desired insight
- Unsupervised learning
  - Relies on unsupervised data
  - The goal is rather to analyze the observed data set
- (Reinforcement learning)
  - Learn to react to the environment
  - Not covered in this course

Exercises: Solutions

- Given a dataset where each sample is represented with a list of meteorological measurements, the goal is to predict the burned area of the corresponding forest fire
  - If you are given the ground-truth burned areas for a set of training samples, what general class of algorithms would you use to solve this problem?
    - Solution: Supervised learning
  - What type of Machine Learning problem is this?
    - Solution: A regression problem (we are predicting a continuous quantity, not a category label)

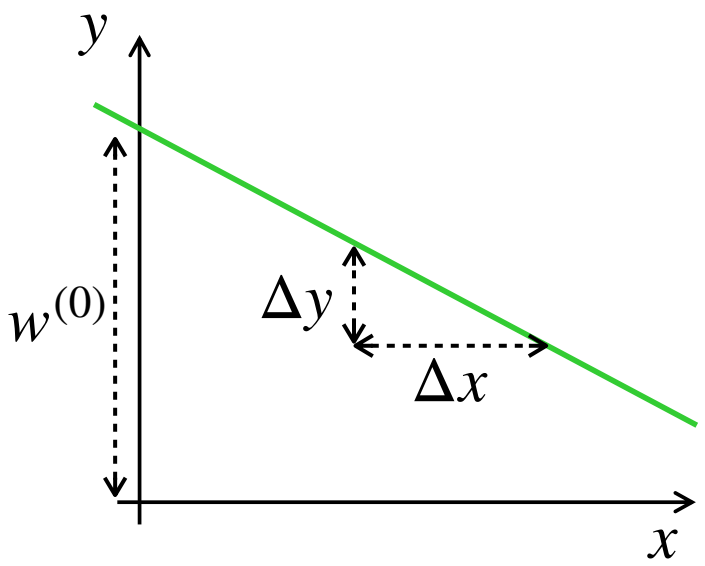
Exercises: Solutions

- Given the following dataset for birth weight prediction:

	Age at delivery	Weight prior to pregnancy (pounds)	Smoker	Doctor visits during 1st trimester	Race	Birth Weight (grams)
Patient 1	29	140	Yes	2	Caucasian	2977
Patient 2	32	132	No	4	Caucasian	3080
Patient 3	36	175	No	0	African-Am	3600
*	*	*	*	*	*	*
*	*	*	*	*	*	*
Patient 189	30	95	Yes	2	Asian	3147

  - How many samples (N) can you assume this dataset to contain?
  - What is the dimensionality (D) of the input to the ML model?
  - What is the dimensionality (C) of the output of the ML model?
    - Solutions: $N = 189,\ D = 5,\ C = 1$

Recap: A simple parametric model: The line

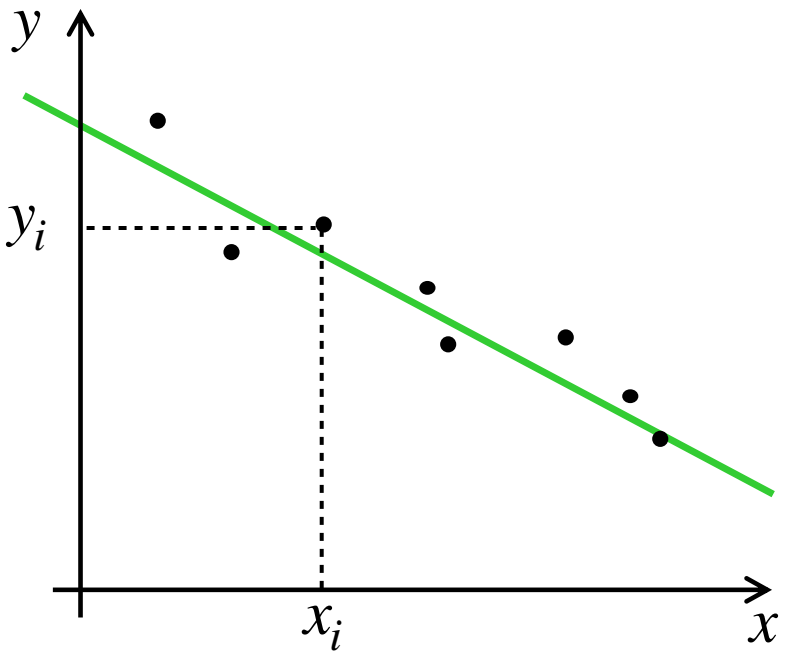


- Defined by 2 parameters
  - The y-intercept $w^{(0)}$
  - The slope $w^{(1)} = \dfrac{\Delta y}{\Delta x}$
- Mathematically, a line is expressed as

$$y = w^{(1)}x + w^{(0)}$$

Recap: Line fitting with noise

- Given N pairs $\{(x_i, y_i)\}$ of noisy measurements, find the line that best fits these observations



- This process is called *linear regression*

Recap: 1D Linear regression: Training

- In essence, fitting a line consists of finding the best line parameters $w^{(0)*}$ and $w^{(1)*}$ for some given data

- This corresponds to the training stage:
  - Given N training pairs $\{(x_i, y_i)\}$, we aim to find $(w^{(0)*}, w^{(1)*})$, such that the predictions of the model

$$\hat{y}_i = w^{(1)*} x_i + w^{(0)*}$$

    are close to the true values y_i

- We then need to define a measure of closeness between y_i and $\hat{y}_i$

Recap: 1D Linear regression: Training

- In practice, one typically uses the squared Euclidean distance

$$d^2(\hat{y}_i, y_i) = (\hat{y}_i - y_i)^2$$

- Training can then be expressed as the *least-squares* minimization problem

$$\min_{w^{(0)}, w^{(1)}} \frac{1}{N} \sum_{i=1}^{N} d^2(\hat{y}_i, y_i)$$

where $\hat{y}_i$ depends on $w^{(0)}$ and $w^{(1)}$

Goals of today's lecture

- Review basic derivatives and gradients concepts

- Derive the solution for linear regression, first with 1D inputs, then with multi-dimensional inputs, finally with multi-dimensional outputs

- Interpret a trained linear model

Minimizing the risk

- The linear regression training problem is a special case of empirical risk minimization
- Minimizing the risk involves computing its derivatives w.r.t. the model parameters
- Let us then review the notion of derivatives/gradient
  - This part is not specific to ML, but it will be useful in the upcoming lectures

Interlude

Derivatives and Gradients

Derivative of a 1-variable function

- The derivative of a function $R(w)$ of a single variable w is the rate at which R changes as w changes

- It is measured for an infinitesimal change in w, starting from a point $\bar{w}$, and written as

$$R'(\bar{w}) = \frac{dR}{dw} = \lim_{\Delta w \to 0} \frac{R(\bar{w} + \Delta w) - R(\bar{w})}{\Delta w}$$

Derivative of a 1-variable function

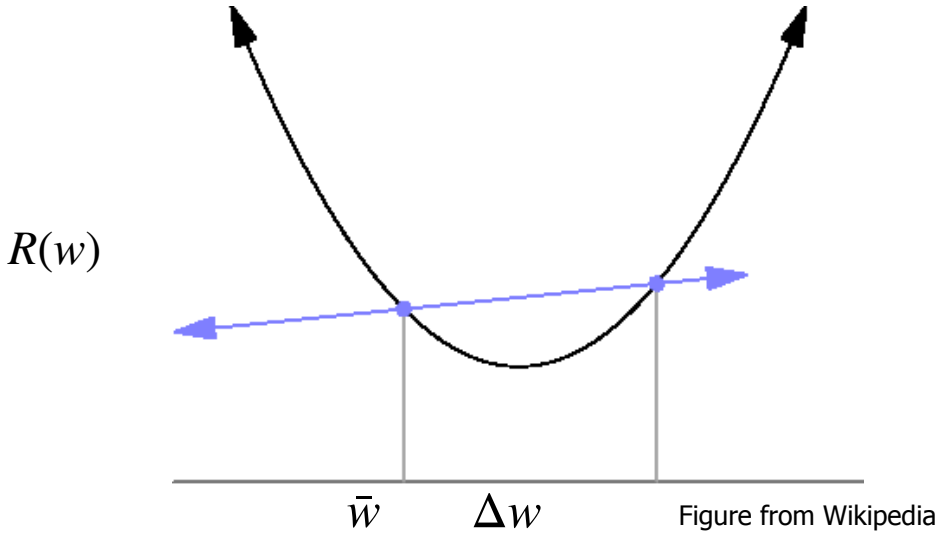


Figure from Wikipedia

- As Δw decreases, the line joining $\bar{w}$ and $\bar{w} + \Delta w$ becomes the tangent to the function
- The derivative is the slope of this tangent

Derivative of a 1-variable function

- Example: More complicated function

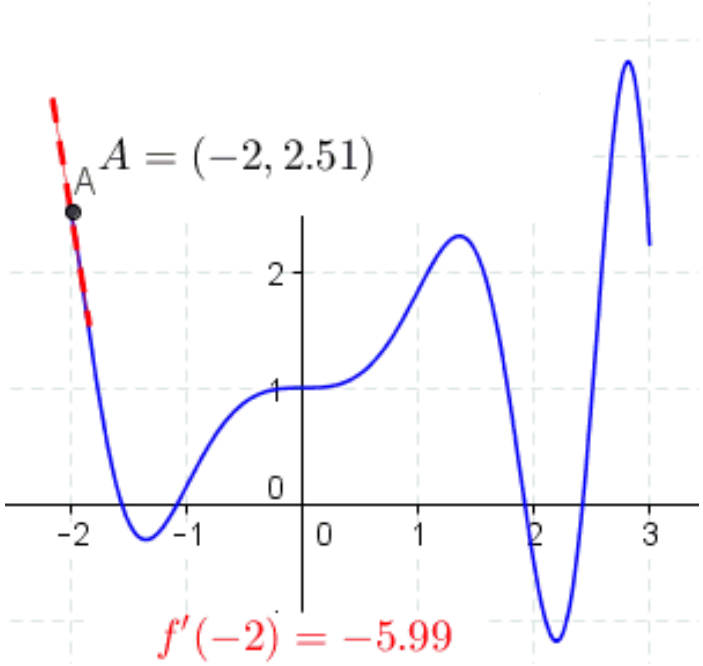
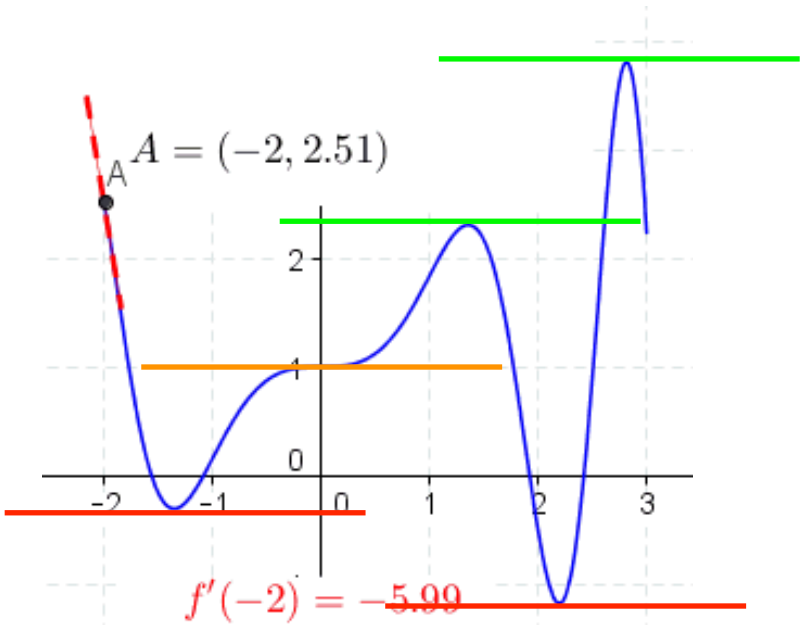


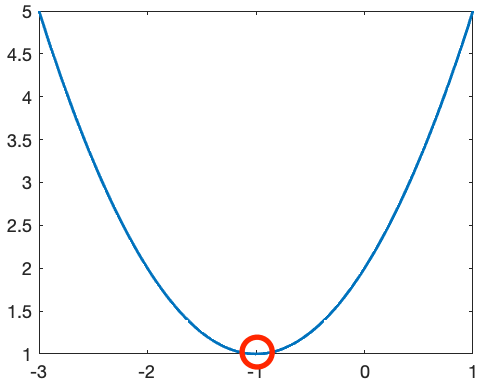
Figure from Wikipedia

Properties of derivatives

- The derivative vanishes at the stationary points:
  - Minima
  - Maxima
  - Saddle points



Minimizing a function



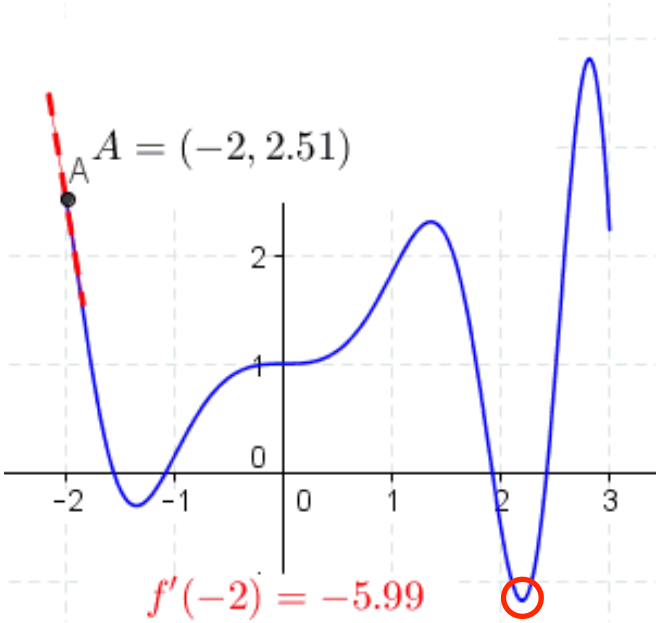
- To minimize a function, we seek a point where the derivative vanishes, i.e., we search for w^* such that

$$\frac{dR(w^*)}{dw} = 0$$

- For some simple, convex functions, w^* can be obtained algebraically by solving the corresponding equation
  - This is referred to as closed-form solution

More complicated functions

- For more complicated, non-convex functions, there are typically multiple, local minima



- Note that, in this example, there is still only one global minimum

Properties of derivatives

- The derivative of a sum (average) is the sum (average) of the derivatives

- In our ML context, this means that the derivative of the empirical risk at parameter $\bar{w}$ can be computed as

$$\frac{dR(\bar{w})}{dw} = \frac{1}{N}\sum_{i=1}^{N}\frac{d\ell\left(\hat{y}_i(\mathbf{x}_i, \bar{w}), y_i\right)}{dw}$$

derivative of the loss for each sample

Dealing with multivariate functions

- In practice, we typically have more than one parameter
  - E.g., even with a single input dimension, linear regression has two parameters ($w^{(0)}$ and $w^{(1)}$)

- To handle this, we rely on the function *gradient*
  - The gradient is a multi-variable generalization of the derivative
  - For a function with D parameters, the gradient is a D-dimensional vector
  - Each element of this vector is the partial derivative of the function w.r.t. a single variable, i.e., the derivative of the function w.r.t. one variable while keeping the other ones constant

Gradient

- Formally, the gradient of a function $R(\mathbf{w})$ with $\mathbf{w} \in \mathbb{R}^D$ is denoted by

$$\nabla R(\mathbf{w}) = \begin{bmatrix} \frac{\partial R}{\partial w^{(1)}} \\ \frac{\partial R}{\partial w^{(2)}} \\ \vdots \\ \frac{\partial R}{\partial w^{(D)}} \end{bmatrix} \in \mathbb{R}^D$$

where $\frac{\partial R}{\partial w^{(j)}}$ denotes the partial derivative w.r.t. variable $w^{(j)}$

Properties of gradient

- The gradient at a point $\mathbf{w}$ has the direction of greatest increase of the function at $\mathbf{w}$
- Its magnitude is the rate of increase in that direction

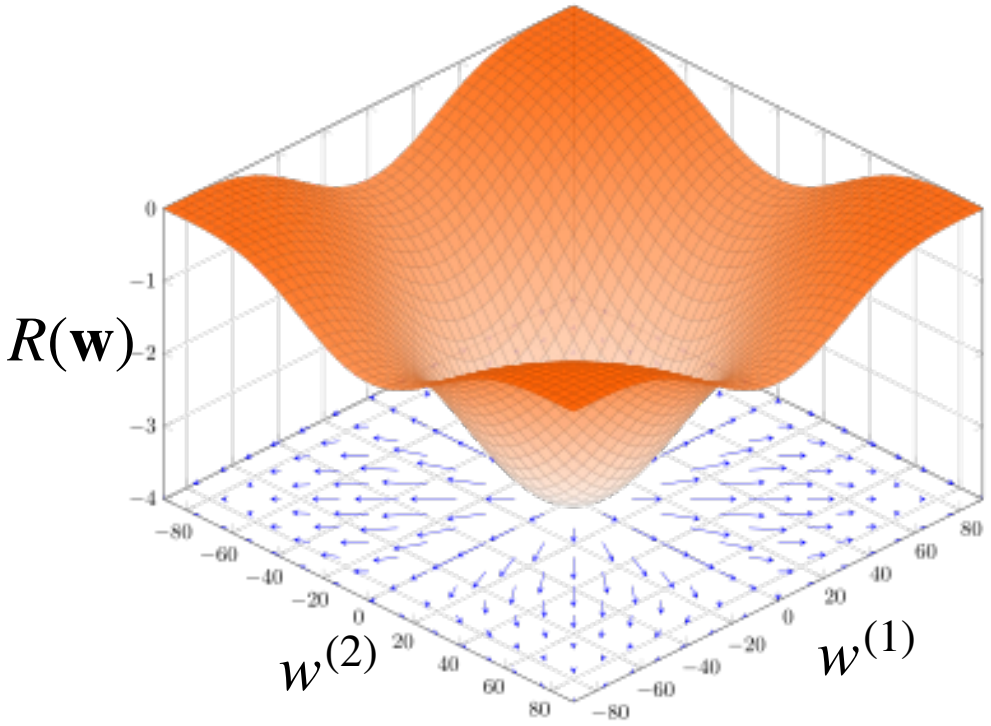
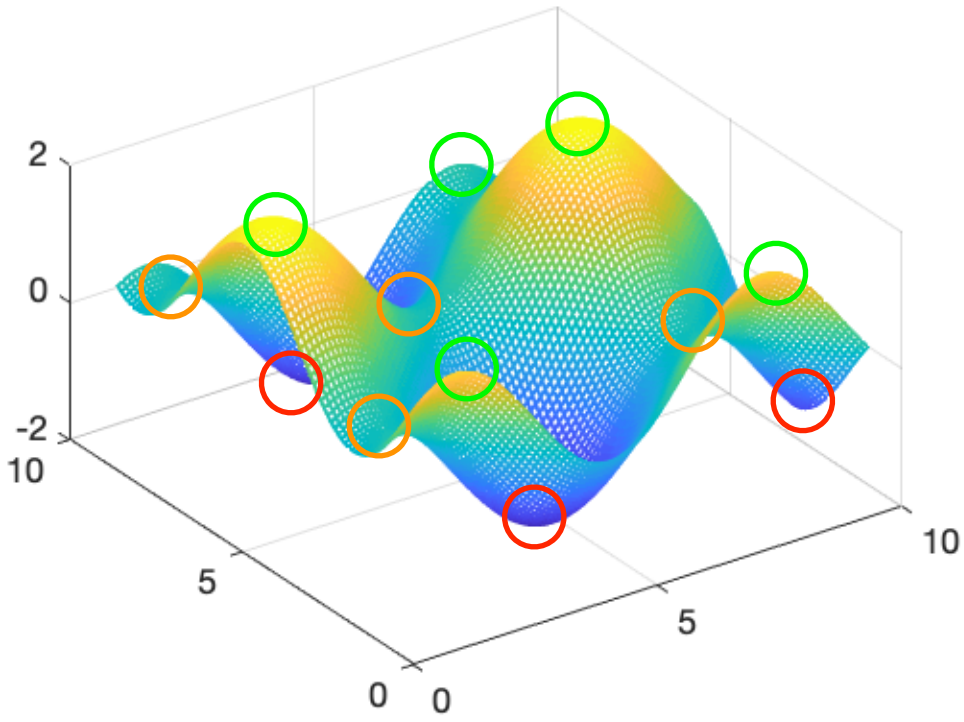


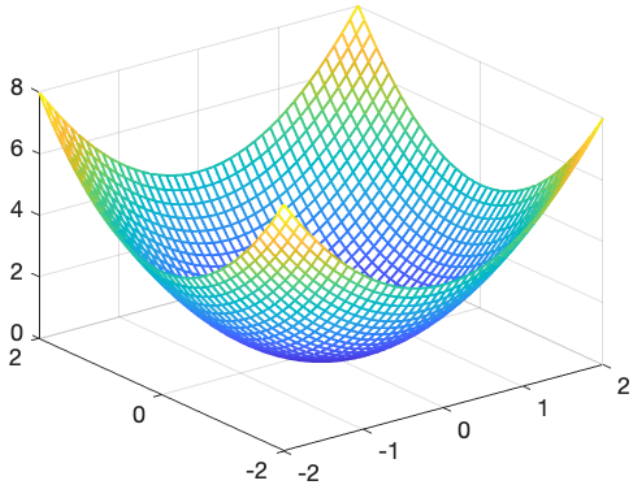
Figure from Wikipedia

Properties of gradient

- The gradient vanishes (becomes a zero vector) at the stationary points of the function
  - Minima
  - Maxima
  - Saddle points



Minimizing a multivariate function



- As in the 1D case, to minimize a function, we seek the point $\mathbf{w}^*$ where the gradient vanishes, i.e.,

$$\nabla R(\mathbf{w}^*) = \mathbf{0}$$

- Recall that the gradient is a vector, so we have a system of equations
- This can still be solved in closed form for some functions

End of the interlude

Back to Linear Regression

1D linear regression: Solution

- With our least-square formulation, our empirical risk is

$$R = \frac{1}{N}\sum_{i=1}^{N} d^2(\hat{y}_i, y_i) = \frac{1}{N}\sum_{i=1}^{N} (w^{(1)}x_i + w^{(0)} - y_i)^2$$

- This gives us the partial derivatives

$$\frac{\partial R}{\partial w^{(0)}} = \frac{2}{N}\sum_{i=1}^{N} (w^{(1)}x_i + w^{(0)} - y_i)$$

$$\frac{\partial R}{\partial w^{(1)}} = \frac{2}{N}\sum_{i=1}^{N} (w^{(1)}x_i + w^{(0)} - y_i) \cdot x_i$$

- We would like to find the values of $w^{(0)}$ and $w^{(1)}$ such that these partial derivatives become 0

1D Linear regression: Solution

- To do this, it is easier to group the parameters $w^{(0)}$ and $w^{(1)}$ into a single parameter vector $\mathbf{w} \in \mathbb{R}^2$

- This lets us re-write our least-square empirical risk as

$$R = \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{w}^T \begin{bmatrix} x_i \\ 1 \end{bmatrix} - y_i \right)^2$$

where the 1 allows us to account for $w^{(0)}$

- We can then define a vector $\mathbf{x}_i \in \mathbb{R}^2$, whose 2nd element is 1, such that

$$R = \frac{1}{N} \sum_{i=1}^{N} \left(\mathbf{x}_i^T \mathbf{w} - y_i \right)^2$$

Side note: Vector-based derivatives

- To compute the gradient of this empirical risk, we can make use of the following general vector derivative rule:
  - Given 2 vectors $(\mathbf{a}, \mathbf{b})$ in $\mathbb{R}^D$

$$\nabla_{\mathbf{b}}(\mathbf{a}^T\mathbf{b}) = \mathbf{a}$$

- Note that useful rules for matrix (or vector) computations can be found in the Matrix Cookbook:
  - http://www2.imm.dtu.dk/pubdb/edoc/imm3274.pdf

1D Linear regression: Solution

- With our empirical risk $R = \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}_i^T\mathbf{w} - y_i\right)^2$

  we can then use the chain rule:

$$\nabla_{\mathbf{w}} R = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial(\mathbf{x}_i^T\mathbf{w} - y_i)^2}{\partial(\mathbf{x}_i^T\mathbf{w} - y_i)} \cdot \frac{\partial(\mathbf{x}_i^T\mathbf{w} - y_i)}{\partial\mathbf{w}}$$

$$= \frac{2}{N}\sum_{i=1}^{N}(\mathbf{x}_i^T\mathbf{w} - y_i) \cdot \mathbf{x}_i$$

- One can easily verify that, because the 2nd element of $\mathbf{x}_i$ is 1, this matches the partial derivatives on Slide 28

1D Linear regression: Solution

- To obtain the solution, we search for $\mathbf{w}^*$ such that

$$\nabla R = \frac{2}{N} \sum_{i=1}^{N} \mathbf{x}_i (\mathbf{x}_i^T \mathbf{w}^* - y_i) = \mathbf{0}$$

(Note that I moved $\mathbf{x}_i$ in front of $(\mathbf{x}_i^T \mathbf{w} - y_i)$, which makes no difference since $(\mathbf{x}_i^T \mathbf{w} - y_i)$ is a scalar)

- This means

$$\left(\sum_{i=1}^{N} \mathbf{x}_i \mathbf{x}_i^T \right) \mathbf{w}^* = \sum_{i=1}^{N} \mathbf{x}_i y_i$$

2×2 matrix $\quad$ 2×1 vector $\quad$ 2×1 vector

(Note that I removed the factor $2/N$, which has no influence on the solution)

1D Linear regression: Matrix form

- Let us now group all $\{\mathbf{x}_i\}$ and all $\{y_i\}$ in a matrix and a vector

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_N^T \end{bmatrix} = \begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \in \mathbb{R}^{N \times 2}$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix} \in \mathbb{R}^N$$

1D Linear regression: Solution

- Then, we can re-write the solution as

$$\mathbf{X}^T\mathbf{X}\mathbf{w}^* = \mathbf{X}^T\mathbf{y}$$

- This finally gives us

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{X}^\dagger\mathbf{y}$$

where $\mathbf{X}^\dagger$ is known as the *Moore-Penrose pseudo-inverse* of $\mathbf{X}$

- In other words, we can obtain the solution to linear regression in closed form

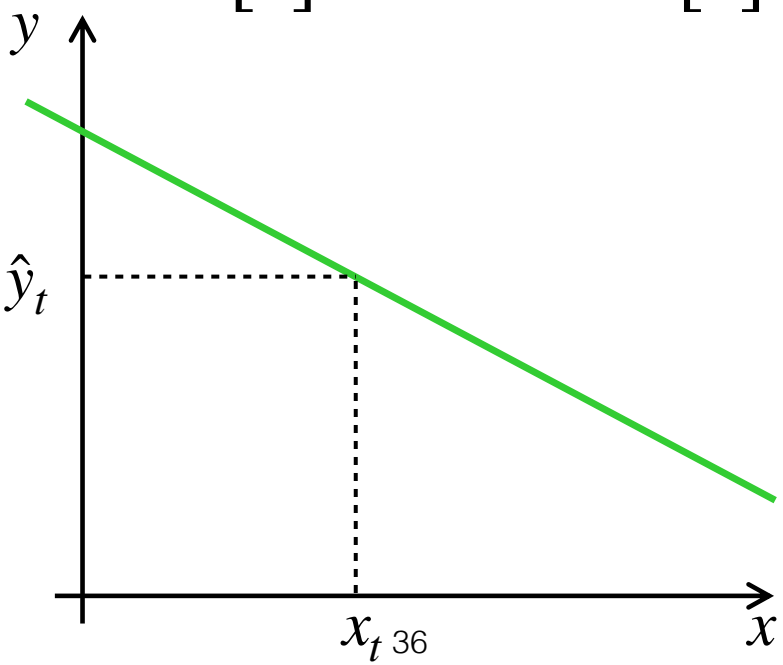
1D linear regression: Demo

- http://digitalfirst.bfwpub.com/stats_applet/stats_applet_5_correg.html

1D Linear regression: Test time

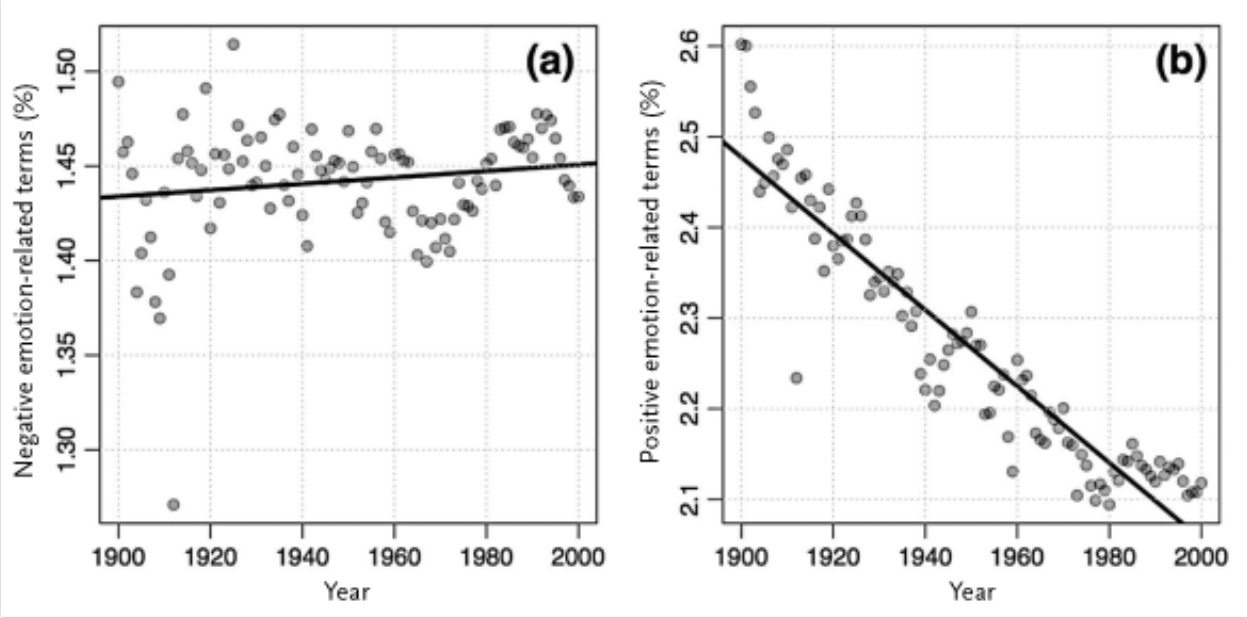
- As mentioned before, once we have the optimal parameters $\mathbf{w}^*$, we can predict $\hat{y}_t$ for any new x_t
- The predicted value is given by

$$\hat{y}_t = \begin{bmatrix} x_t \\ 1 \end{bmatrix}^T \mathbf{w}^* = (\mathbf{w}^*)^T \begin{bmatrix} x_t \\ 1 \end{bmatrix}$$



1D linear regression: Test time

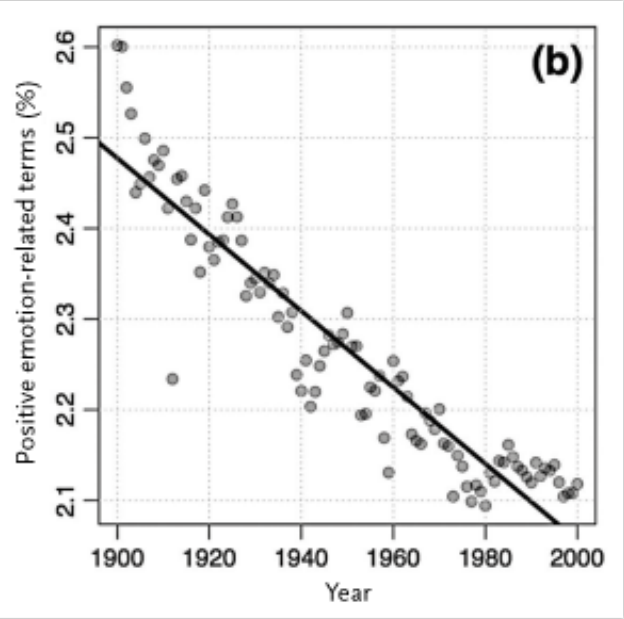
- Discover trends:
  - Example: Proportion of negative and positive emotions in anglophone fiction (Morin & Acerbi, 2016. Figure from Moretti & Sobchuk, 2019)



  - With such temporal trends, one can predict what will happen in the future

Exercises

- From the trend data below



- Give an example of what numerical values one $\mathbf{x}_i$ and the corresponding y_i could roughly take
- What would the matrix $\mathbf{X}$ and the vector $\mathbf{y}$ look like (using rough numerical values for a few samples)?

Dealing with multiple input dimensions

- In general, an input observation $\mathbf{x}_i$ is not represented by a single value
  - E.g., a grayscale image can be represented by an $W \cdot H$ dimensional vector

$$\mathbf{x}_i = \text{vectorize}(\text{  }) \in \mathbb{R}^{28 \cdot 28} = \mathbb{R}^{784}$$

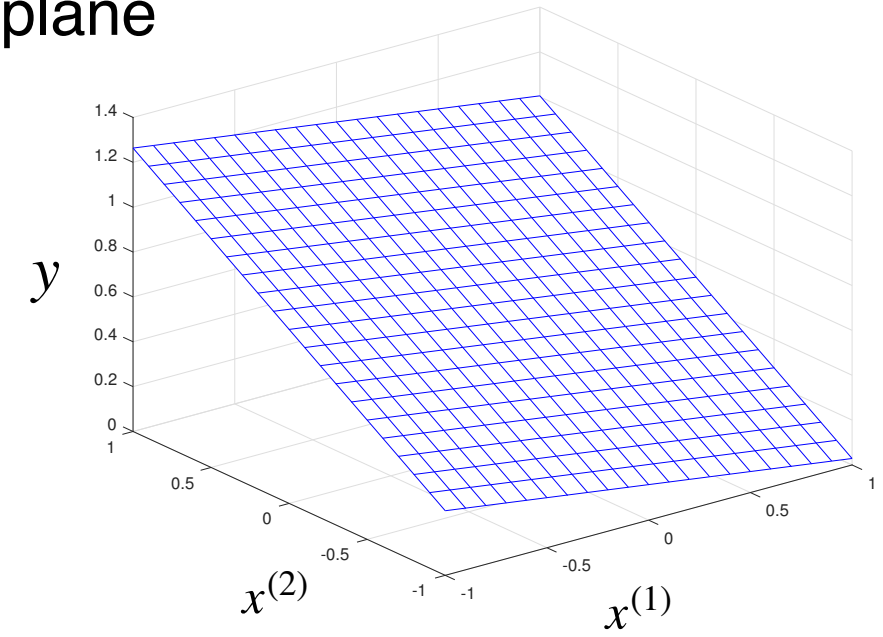
  - E.g., with attribute-based representations, multiple attributes are often given

Birth weight prediction

	Age at delivery	Weight prior to pregnancy (pounds)	Smoker	Doctor visits during 1st trimester	Race	Birth Weight (grams)
Patient 1	29	140	Yes	2	Caucasian	2977
Patient 2	32	132	No	4	Caucasian	3080
Patient 3	36	175	No	0	African-Am	3600
*	*	*	*	*	*	*
*	*	*	*	*	*	*
Patient 189	30	95	Yes	2	Asian	3147

Plane

- When there are two input dimensions, instead of a line, we can define a plane

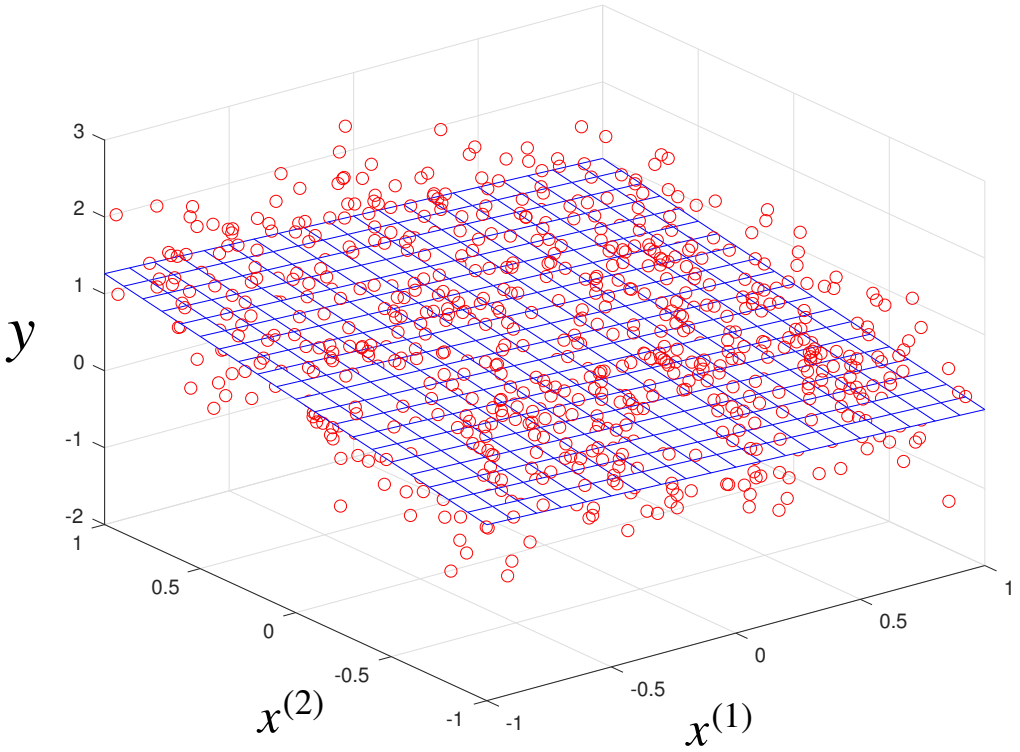


- Mathematically, a plane is expressed as

$$y = w^{(0)} + w^{(1)}x^{(1)} + w^{(2)}x^{(2)} = \mathbf{w}^T \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ 1 \end{bmatrix}$$

Plane fitting

- Given N noisy pairs $\{(\mathbf{x}_i, y_i)\}$, where $\mathbf{x}_i \in \mathbb{R}^2$, find the plane that best fits these observations



Hyperplane

- This can be generalized to higher dimensions
- In dimension D, we can write

$$y = w^{(0)} + w^{(1)}x^{(1)} + w^{(2)}x^{(2)} + \ldots + w^{(D)}x^{(D)} = \mathbf{w}^T \begin{bmatrix} x^{(1)} \\ x^{(2)} \\ \vdots \\ x^{(D)} \\ 1 \end{bmatrix}$$

- Ultimately, whatever the dimension, we can write

$$y = \mathbf{w}^T\mathbf{x}$$

with $\mathbf{x} \in \mathbb{R}^{D+1}$, where the extra dimension contains a 1 to account for $w^{(0)}$

Multi-input linear regression: Training

- Because the output remains 1D, we can use the same least-square loss function as before, and write training as

$$\min_{\mathbf{w}} \sum_{i=1}^{N} d^2(\hat{y}_i, y_i) \Leftrightarrow \min_{\mathbf{w}} \sum_{i=1}^{N} (\mathbf{x}_i^T \mathbf{w} - y_i)^2$$

- Note that this has the same form as in the 1D input case when we grouped $w^{(0)}$ and $w^{(1)}$ in a single vector
  - And I removed the $1/N$ factor in front of the sum, which, as shown before, is unnecessary because it does not affect the solution

Multi-input linear regression: Solution

- Therefore the solution, obtained by zeroing out the gradient of the empirical risk, is exactly the same as before

$$\mathbf{w}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{y} = \mathbf{X}^\dagger\mathbf{y}$$

- But now, the matrix $\mathbf{X} \in \mathbb{R}^{N\times(D+1)}$, instead of $\mathbb{R}^{N\times 2}$ before

- Note that this makes perfect sense:
  - Before, we had 1D inputs, and thus $D = 1$ (and so $D + 1 = 2$)

Linear regression: Demo

- https://playground.tensorflow.org/#activation=linear&batchSize=10&dataset=circle&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=0&networkShape=&seed=0.45772&showTestData=false&discretize=false&percTrainData=50&x=true&y=true&xTimesY=false&xSquared=false&ySquared=false&cosX=false&sinX=false&cosY=false&sinY=false&collectStats=false&problem=regression&initZero=false&hideText=false

Interlude

Interpreting a Linear Model

Linear regression: Example

- UCI Wine Quality dataset:
  - Predict the quality of wine based on several attributes (5 samples shown below)

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

- Final RMSE:
  - 0.65 for training data
  - 0.63 for test data

Example from https://medium.com/datadriveninvestor/regression-from-scratch-wine-quality-prediction-d61195cb91c8

Linear regression: Example

- One can then look at the coefficient values (i.e., the $\{w^{(i)}\}$) to see the influence of each attribute

	Coeffecient
fixed acidity	0.017737
volatile acidity	-0.992560
citric acid	-0.139629
chlorides	-1.590943
free sulfur dioxide	0.005597
total sulfur dioxide	-0.003520
density	0.768590
pH	-0.437414
sulphates	0.812888
alcohol	0.301484

Example from https://medium.com/datadriveninvestor/regression-from-scratch-wine-quality-prediction-d61195cb91c8

Linear regression: Example

- Author age prediction from text
  - N'guyen et al., Workshop on Language Technology for Cultural Heritage, Social Sciences, and Humanities, 2011
  - Influence of features (words)

Younger people

like	-1.295
gender-male	-0.539
LIWC-School	-0.442
just	-0.354
LIWC-Anger	-0.303
LIWC-Cause	-0.290
mom	-0.290
so	-0.271
definitely	-0.263
LIWC-Negemo	-0.256

Older people

years	0.601
POS - dt	0.485
LIWC - Incl	0.483
POS - prp vbp	0.337
granddaughter	0.332
grandchildren	0.293
had	0.277
daughter	0.272
grandson	0.245
ah	0.243

Linear regression: Example

- Example: True age 17, predicted age 16.48

> “I can’t sleep, but this time I have school tommorow, so I have to try I guess. My parents got all pissed at me today because I forgot how to do the homework [...]. Really mad, I ended it pissing off my mom and [...] NOTHING! Damn, when I’m at my cousin’s I have no urge to use the computer like I do here, […]"

- Example: True age 70, predicted age 71.53

> “[...] I was a little bit fearful of having surgery on both sides at once (reduction and lift on the right, tissue expander on the left) [...] On the good side, my son and family live near the plastic surgeon’s office and the hospital, [...], at least from my son and my granddaughter […]"

Interpreting a linear model

- Warning: The magnitude of a coefficient will depend on the magnitude of the corresponding feature/attribute
  - A coefficient might be very small simply to compensate for the fact that the range of the feature is very large
  - E.g., looking at 2 attributes from the UCI Wine Quality example

chlorides	free sulfur dioxide
0.076	11.0
0.098	25.0
0.092	15.0

- This can be addressed by normalizing the data
  - We will discuss this in an upcoming lecture

End of the interlude

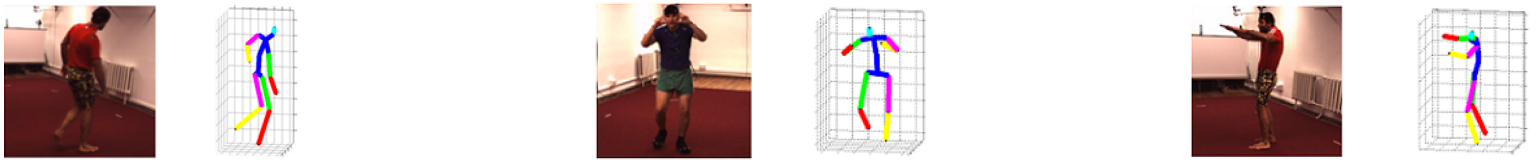
Let's continue with linear regression but now with multiple *output* dimensions

Dealing with multiple output dimensions

- Until now, we have assumed that the output was a single value, i.e., $y_i \in \mathbb{R}$

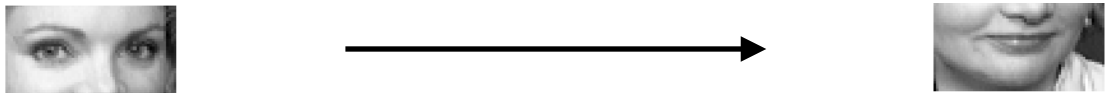
- In practice, however, one may want to output multiple values for a given input, i.e., $\mathbf{y}_i \in \mathbb{R}^C$, with $C > 1$

- E.g., human pose estimation
  - We aim to predict the 3D position of each joint, i.e., for J joints, $C = 3J$



Multi-output linear regression: Example

- Face completion:
  - Example from https://scikit-learn.org/stable/auto_examples/miscellaneous/plot_multioutput_face_completion.html#sphx-glr-auto-examples-miscellaneous-plot-multioutput-face-completion-py
  - Task: Given the top half image of a face, predict the bottom half



Input Model (e.g., linear regression) Output

  - Dataset: Olivetti faces
    - 40 subjects (35 for training, 5 for testing)
    - 10 images per subject

Multi-output linear regression: Example

- Face completion:
  - Results:

Multi-output linear regression: Model

- To output multiple values, the linear model cannot just rely on a vector $\mathbf{w}$, because the product $\mathbf{w}^T\mathbf{x}_i$ yields a single value

- We therefore use a matrix $\mathbf{W} \in \mathbb{R}^{(D+1)\times C}$, such that

$$\hat{\mathbf{y}}_i = \mathbf{W}^T\mathbf{x}_i = \begin{bmatrix} \mathbf{w}_{(1)}^T \\ \mathbf{w}_{(2)}^T \\ \vdots \\ \mathbf{w}_{(C)}^T \end{bmatrix} \mathbf{x}_i$$

where each $\mathbf{w}_{(j)}$ is a $(D+1)$-dimensional vector, used to predict one output dimension

Multi-output linear regression: Training

- Since the outputs are multi-dimensional, we need to slightly modify the loss function

- The multi-dimensional counterpart of the 1D least-square loss is the squared Euclidean distance

- So we can write multi-output linear regression as

$$\min_{\mathbf{W}} \sum_{i=1}^{N} \|\mathbf{W}^T \mathbf{x}_i - \mathbf{y}_i\|^2$$

where $\|\mathbf{a}\| = \sqrt{\sum_{j=1}^{C} \left(a^{(j)}\right)^2}$ indicates the norm of a vector (here $\mathbf{a} \in \mathbb{R}^C$)

Multi-output linear regression: Gradient

- To compute the gradient of this function, we make use of the following two properties:

1. Norm and derivative:

$$\|\mathbf{a}\|^2 = \mathbf{a}^T\mathbf{a}$$

and thus (by the vector derivative we have seen before)

$$\nabla_{\mathbf{a}}\|\mathbf{a}\|^2 = 2\mathbf{a}$$

2. Matrix derivative:

$$\nabla_{\mathbf{B}}(\mathbf{a}^T\mathbf{B}) = \mathbf{a}$$

Multi-output linear regression: Gradient

- Thus, using the chain rule (and replacing $(\mathbf{W}^T\mathbf{x}_i - \mathbf{y}_i)$ with $(\mathbf{x}_i^T\mathbf{W} - \mathbf{y}_i^T)$), we obtain the gradient

$$\nabla_{\mathbf{W}} R = 2\sum_{i=1}^{N} \mathbf{x}_i \left(\mathbf{x}_i^T\mathbf{W} - \mathbf{y}_i^T\right)$$

- Setting it to zero means that

$$\left(\sum_{i=1}^{N} \mathbf{x}_i\mathbf{x}_i^T\right)\mathbf{W}^* = \sum_{i=1}^{N} \mathbf{x}_i\mathbf{y}_i^T$$

$(D+1)\times(D+1)$ matrix $(D+1)\times C$ matrix $(D+1)\times C$ matrix

Multi-output linear regression: Solution

- We can again group the inputs in a matrix $\mathbf{X} \in \mathbb{R}^{N \times (D+1)}$, but the outputs now need to be grouped in a matrix (not a vector anymore), as

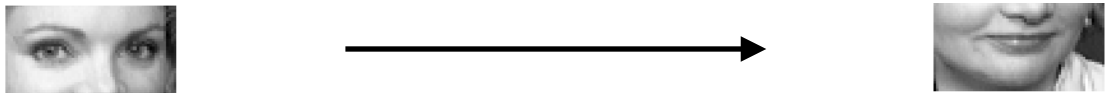
$$\mathbf{Y} = \begin{bmatrix} \mathbf{y}_1^T \\ \mathbf{y}_2^T \\ \vdots \\ \mathbf{y}_N^T \end{bmatrix} = \begin{bmatrix} y_1^{(1)} & y_1^{(2)} & \cdots & y_1^{(C)} \\ y_2^{(1)} & y_2^{(2)} & \cdots & y_2^{(C)} \\ \vdots & \vdots & \vdots & \vdots \\ y_N^{(1)} & y_N^{(2)} & \cdots & y_N^{(C)} \end{bmatrix} \in \mathbb{R}^{N \times C}$$

- Then, following the same strategy as before, we have

$$\mathbf{W}^* = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T\mathbf{Y} = \mathbf{X}^\dagger\mathbf{Y}$$

Exercise

- Given the following face completion task:
  - Task: Given the top half image of a face, predict the bottom half



Input Linear regression Output

  - Assuming that there are 35 training subjects with 10 images per subject, and that a complete face image is of size 28×28 pixels, what is the shape of the matrices $\mathbf{X}$ and $\mathbf{Y}$?
  - How many parameters are there to learn?

Survey

- Please fill in the survey on the Moodle page to comment on the pace of the lecture