CPSC 340: Machine Learning and Data Mining

Least Squares

Fall 2019

Admin

- Assignment 2:
  - 2 late days to hand in tonight, 1 for Wednesday.

- Assignment 3 is up:
  - Start early, this is usually the longest assignment.
  - Has details about the project.
  - Only 1 late day allowed.

- We're going to start using calculus and linear algebra a lot.
  - You should start reviewing these ASAP if you are rusty.
  - A review of relevant calculus concepts is here.
  - A review of relevant linear algebra concepts is here.

Supervised Learning Round 2: Regression

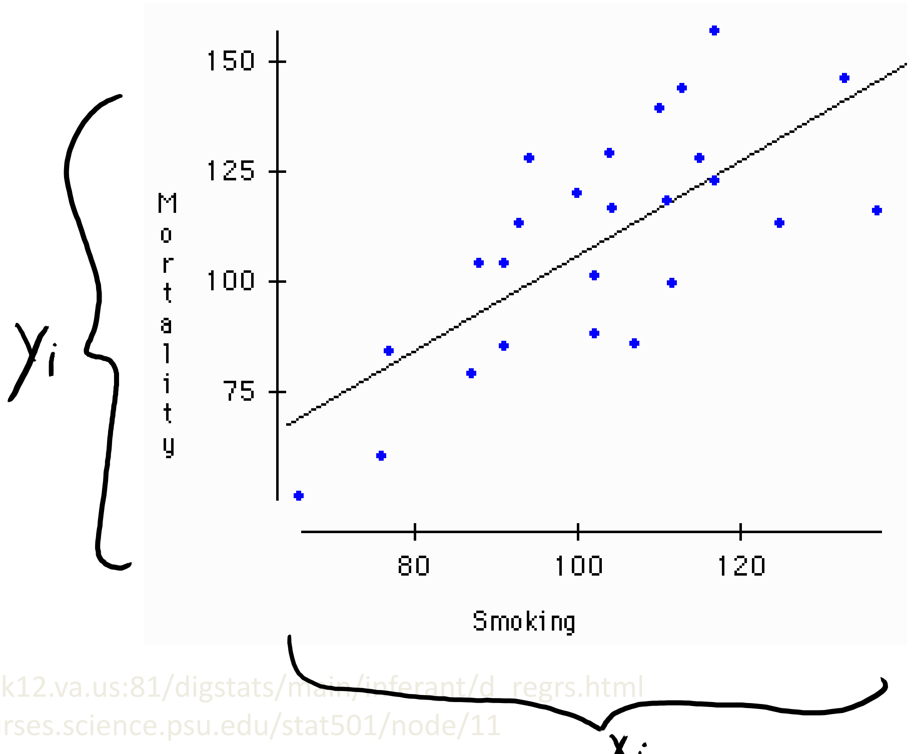
- We're going to revisit supervised learning:

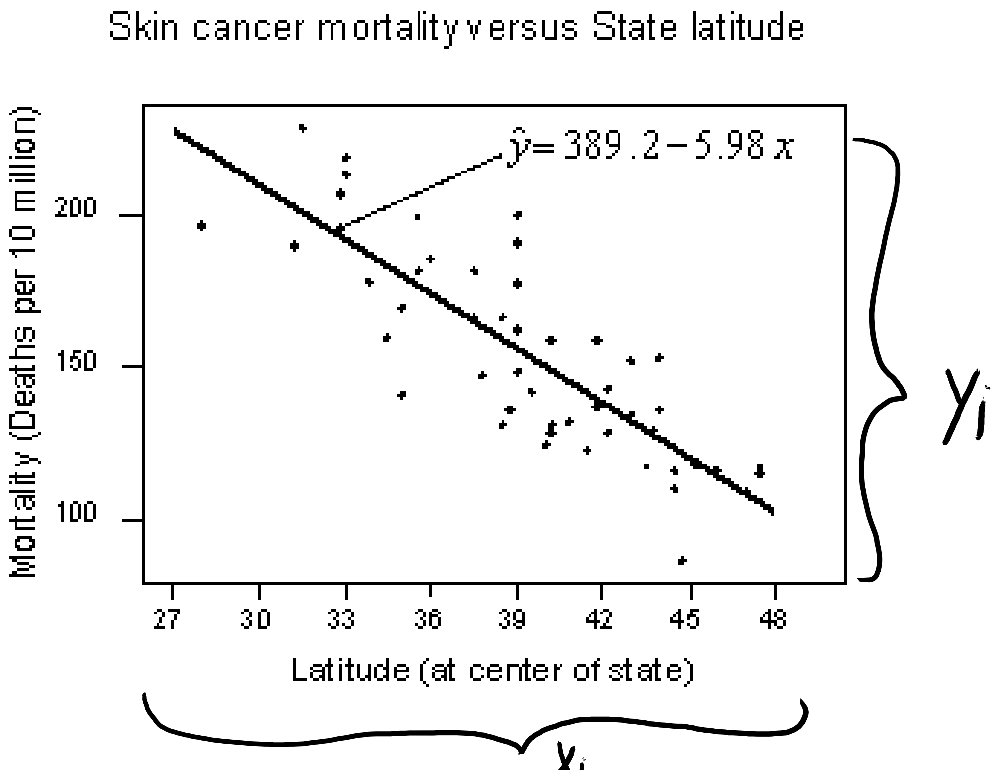
$$X = \begin{bmatrix} \quad & \quad & \quad \end{bmatrix} \quad y = \begin{bmatrix} \; \end{bmatrix}$$

- Previously, we considered classification:
  - We assumed y_i was discrete: y_i = 'spam' or y_i = 'not spam'.
- Now we're going to consider regression:
  - We allow y_i to be numerical: $y_i = 10.34$cm.

Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables:
  - Does number of lung cancer deaths change with number of cigarettes?
  - Does number of skin cancer deaths change with latitude?

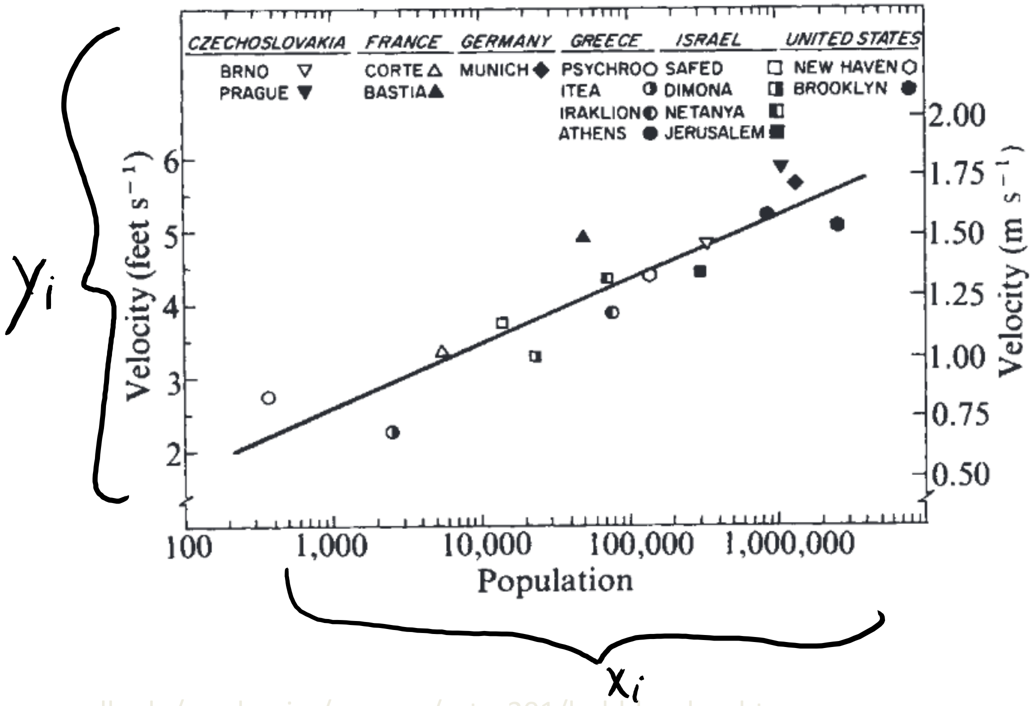


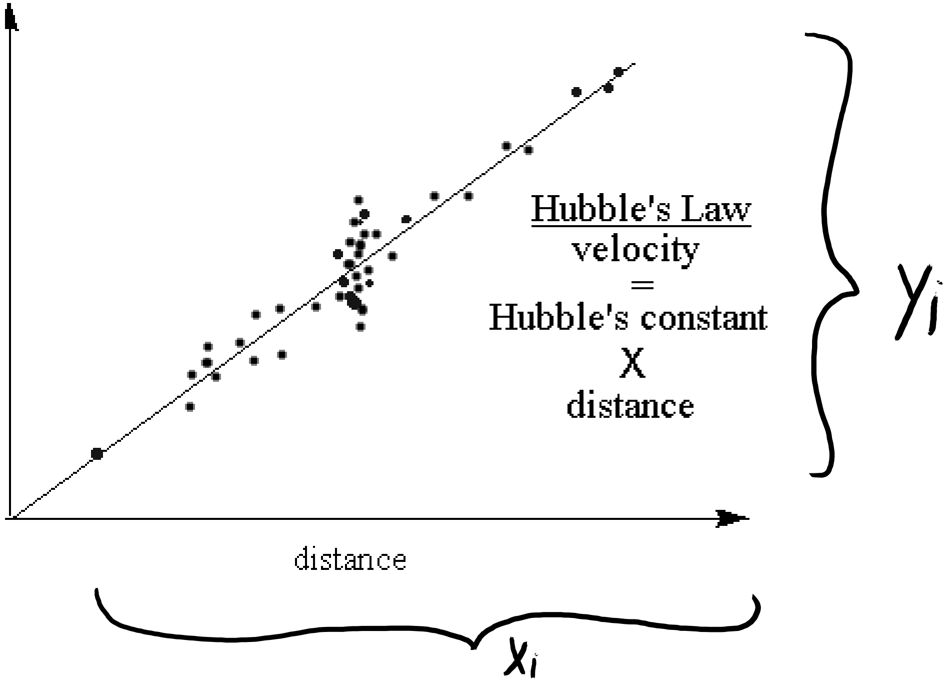


http://www.cvgs.k12.va.us:81/digstats/main/inferant/d_regrs.html
https://onlinecourses.science.psu.edu/stat501/node/11

Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables:
  - Do people in big cities walk faster?
  - Is the universe expanding or shrinking or staying the same size?

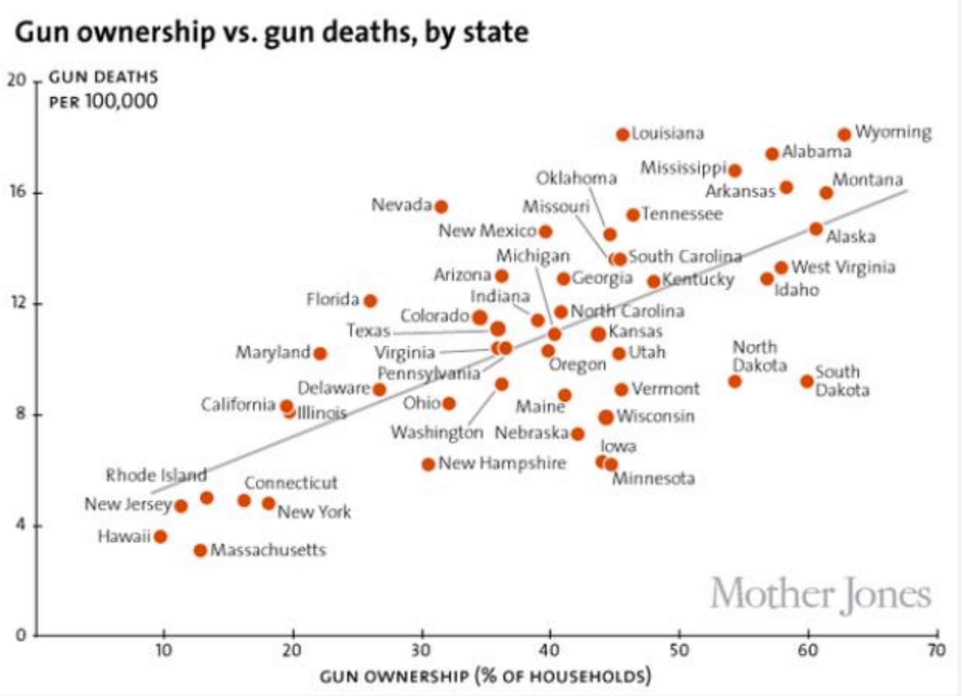


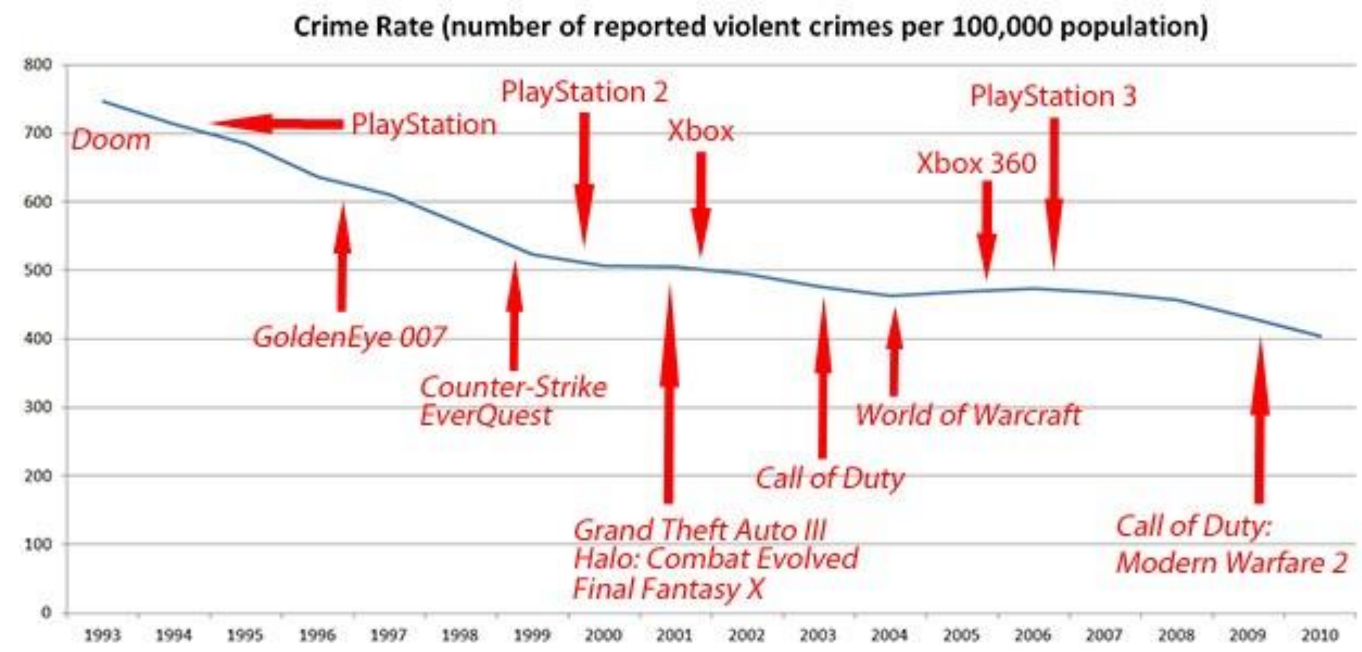


http://hosting.astro.cornell.edu/academics/courses/astro201/hubbles_law.htm
https://www.nature.com/articles/259557a0.pdf

Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables:
  - Does number of gun deaths change with gun ownership?
  - Does number violent crimes change with violent video games?



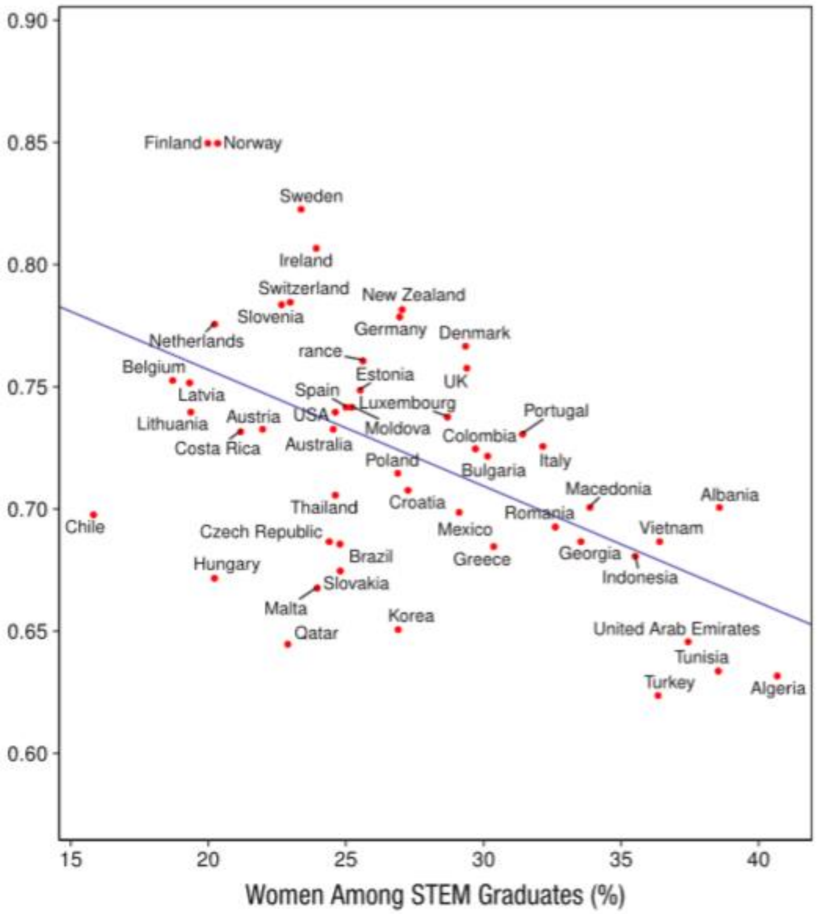


http://www.vox.com/2015/10/3/9444417/gun-violence-united-states-america
https://www.soundandvision.com/content/violence-and-video-games

Example: Dependent vs. Explanatory Variables

- We want to discover relationship between numerical variables:
  - Does higher gender equality index lead to more women STEM grads?

- Not that we're doing supervised learning:
  - Trying to predict value of 1 variable (the 'y_i' values). (instead of measuring correlation between 2).
- Supervised learning does not give causality:
  - OK: "Higher index is correlated with lower grad %".
  - OK: "Higher index helps predict lower grad %".
  - BAD: "Higher index leads to lower grads %".
    - People/media get these confused all the time, be careful!
    - There are lots of potential reasons for this correlation.



https://www.weforum.org/agenda/2018/02/does-gender-equality-result-in-fewer-female-stem-grads/

Handling Numerical Labels

- One way to handle numerical y_i: discretize.
  - E.g., for 'age' could we use {'age $\leq$ 20', '20 < age $\leq$ 30', 'age > 30'}.
  - Now we can apply methods for classification to do regression.
  - But coarse discretization loses resolution.
  - And fine discretization requires lots of data.
- There exist regression versions of classification methods:
  - Regression trees, probabilistic models, non-parametric models.
- Today: one of oldest, but still most popular/important methods:
  - Linear regression based on squared error.
  - Interpretable and the building block for more-complex methods.

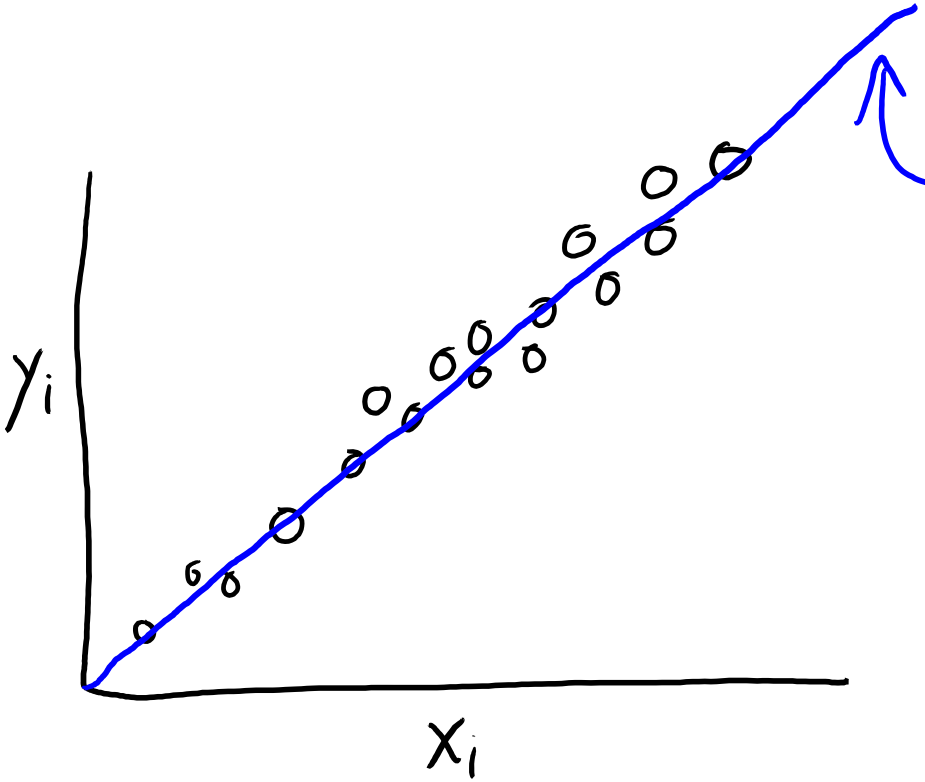
Linear Regression in 1 Dimension

- Assume we only have 1 feature (d = 1):
  - E.g., x_i is number of cigarettes and y_i is number of lung cancer deaths.
- Linear regression makes predictions $\hat{y}_i$ using a linear function of x_i:

$$\hat{y}_i = w x_i$$

- The parameter 'w' is the weight or regression coefficient of x_i.
  - We're temporarily ignoring the y-intercept.
- As x_i changes, slope 'w' affects the rate that $\hat{y}_i$ increases/decreases:
  - Positive 'w': $\hat{y}_i$ increase as x_i increases.
  - Negative 'w': $\hat{y}_i$ decreases as x_i increases.

Linear Regression in 1 Dimension



line $\hat{y}_i = wx_i$ for a particular slope 'w'.

Aside: terminology woes

- Different fields use different terminology and symbols.
  - Data points = **objects** = **examples** = rows = observations.
  - **Inputs** = predictors = **features** = explanatory variables= regressors = independent variables = covariates = columns.
  - **Outputs** = outcomes = targets = response variables = dependent variables (also called a "label" if it's categorical).
  - Regression coefficients = **weights** = parameters = betas.
- With linear regression, the symbols are inconsistent too:
  - In ML, the data is X and the weights are w.
  - In statistics, the data is X and the weights are β.
  - In optimization, the data is A and the weights are x.

Least Squares Objective

- Our linear model is given by:

$$\hat{y}_i = w x_i$$

- So we make predictions for a new example by using:

$$\hat{y}_i = w \tilde{x}_i$$

- But we can't use the same error as before:
  - Even if data comes from a linear model but has noise, we can have $\hat{y}_i \neq y_i$ for all training examples 'i' for the "best" model
  - Unlikely that line goes exactly through the training examples.

Least Squares Objective

- We need a way to evaluate numerical error.
- Classic way is setting slope ‘w’ to minimize sum of squared errors:

$$f(w) = \sum_{i=1}^{n} (w x_i - y_i)^2$$

True value of y_i

Our prediction $\hat{y}_i$

Sum up the squared differences over all training examples.

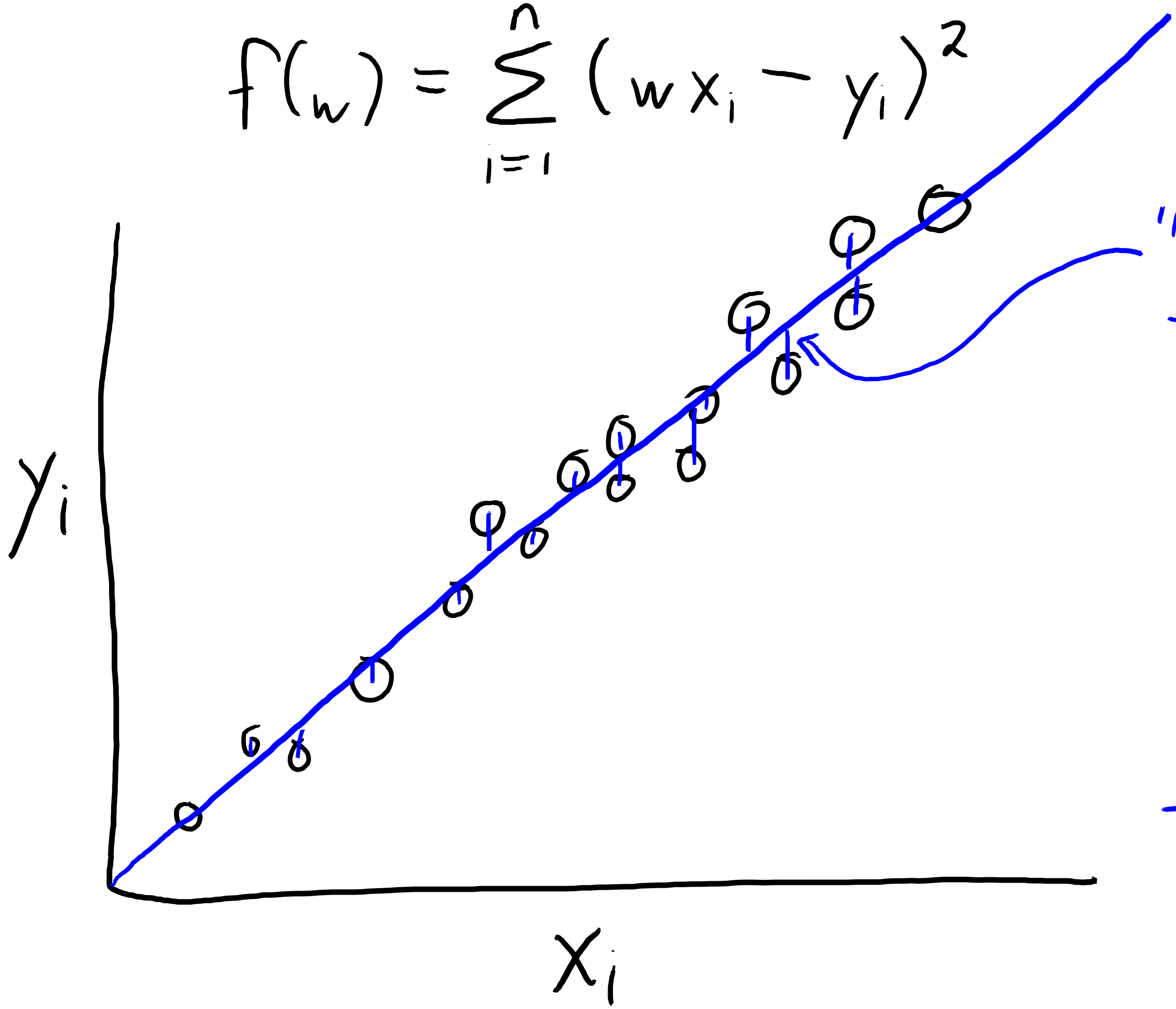
Difference between prediction and true value for example 'i'.

- There are some justifications for this choice.
  – A probabilistic interpretation is coming later in the course.
- But usually, it is done because it is easy to minimize.

Least Squares Objective

- Classic way to set slope 'w' is minimizing sum of squared errors:

$$f(w) = \sum_{i=1}^{n} (w x_i - y_i)^2$$



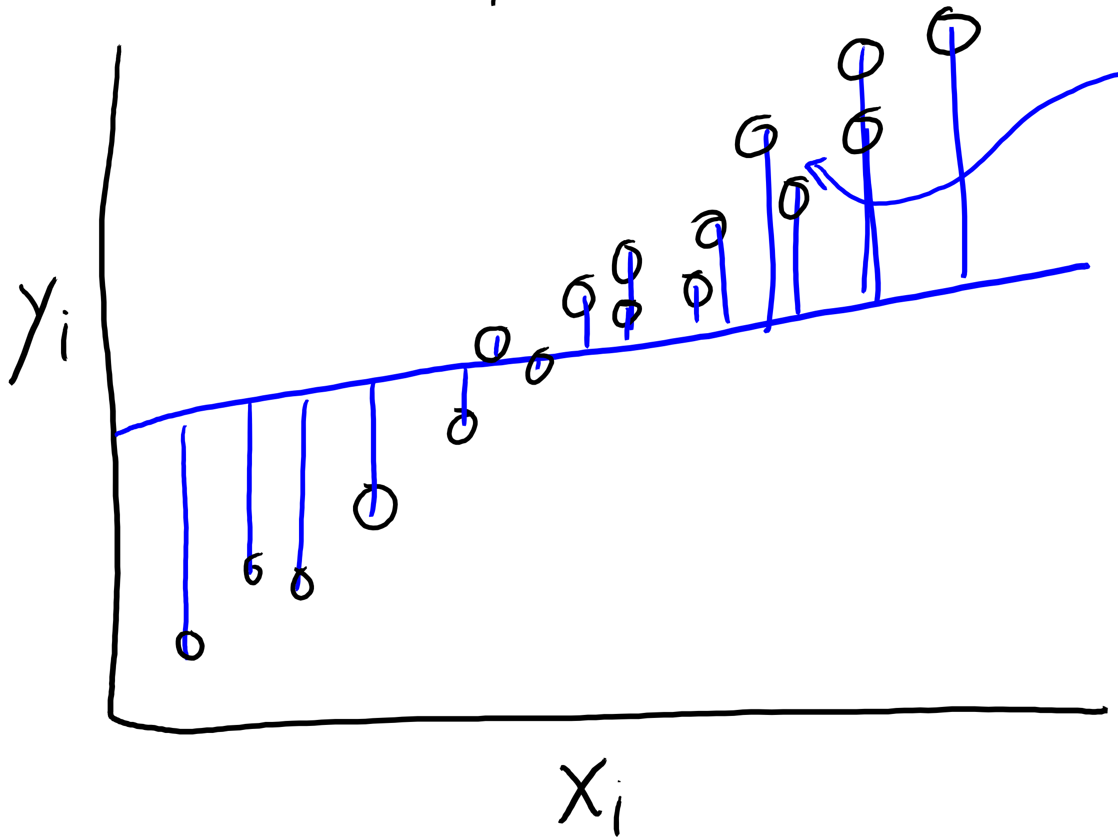
"Error" is the sum of the squared values of these vertical distances between the line (wx_i) and the targets (y_i)

If this error is small, then our predictions are close to the targets.

Least Squares Objective

- Classic way to set slope 'w' is minimizing sum of squared errors:

$$f(w) = \sum_{i=1}^{n} (wx_i - y_i)^2$$

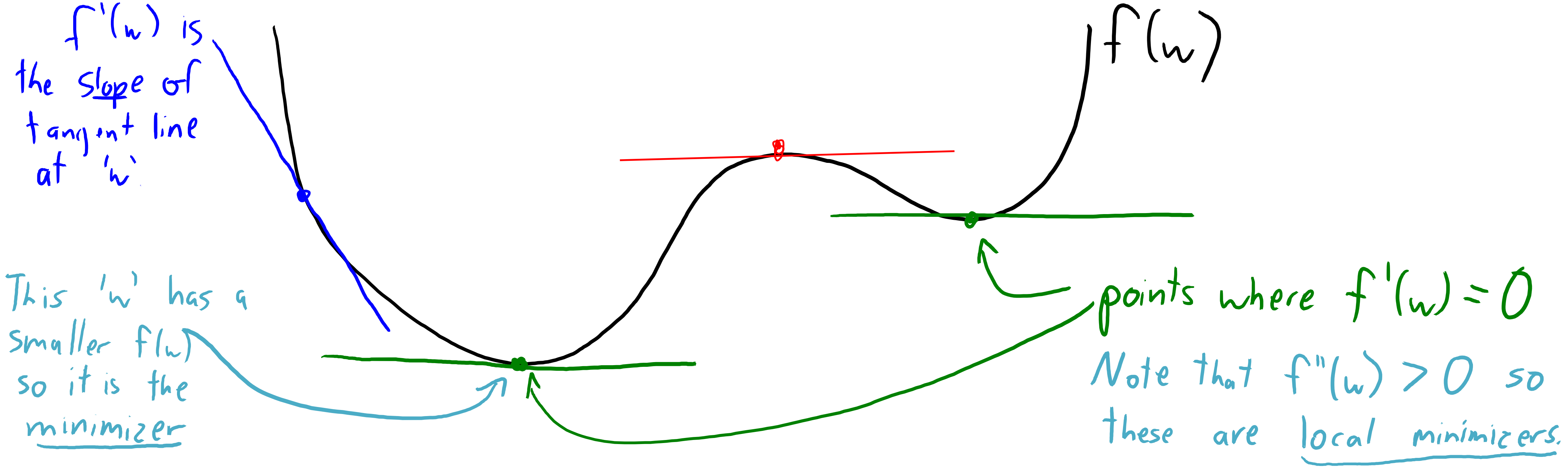


"Error" is the sum of the squared values of these vertical distances between the line (wx_i) and the targets (y_i)

If this error is large, then our predictions are far from the targets.

Minimizing a Differential Function

- Math 101 approach to minimizing a differentiable function ‘f’:
  1. Take the derivative of ‘f’.
  2. Find points ‘w’ where the derivative f’(w) is equal to 0.
  3. Choose the smallest one (and check that f’’(w) is positive).



Digression: Multiplying by a Positive Constant

- Note that this problem:

$$f(w) = \sum_{i=1}^{n} (wx_i - y_i)^2$$

- Has the same set of minimizers as this problem:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n} (wx_i - y_i)^2$$

- And these also have the same minimizers:

$$f(w) = \frac{1}{n}\sum_{i=1}^{n} (wx_i - y_i)^2 \qquad f(w) = \frac{1}{2n}\sum_{i=1}^{n} (wx_i - y_i)^2 + 1000$$

- I can multiply 'f' by any positive constant and not change solution.
  - Gradient will still be zero at the same locations.
  - We'll use this trick a lot!

(Quora trolling on ethics of this)

Finding Least Squares Solution

- Finding 'w' that minimizes sum of squared errors:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(wx_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{n}\left[w^2 x_i - 2wx_iy_i + y_i^2\right]$$

(expand square)

$$= \frac{w^2}{2}\underbrace{\sum_{i=1}^{n} x_i^2}_{\text{constant 'a'}} - w\underbrace{\sum_{i=1}^{n} x_i y_i}_{\text{constant 'b'}} + \underbrace{\frac{1}{2}\sum_{i=1}^{n} y_i^2}_{\text{constant 'c'}}$$

(split sums, take 'w' outside)

$$= \frac{w^2}{2}a - wb + c$$

Take derivative: $f'(w) = wa - b + 0$

Setting $f'(w) = 0$ and solving gives $w = \frac{b}{a} = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$ (exists if we have a non-zero feature)

Finding Least Squares Solution

- Finding ‘w’ that minimizes sum of squared errors:

Setting $f'(w)=0$ and solving gives $w = \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}$ (exists if we have one non-zero x_{ij})

- Let’s check that this is a minimizer by checking second derivative:

$$f'(w) = w\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i y_i$$

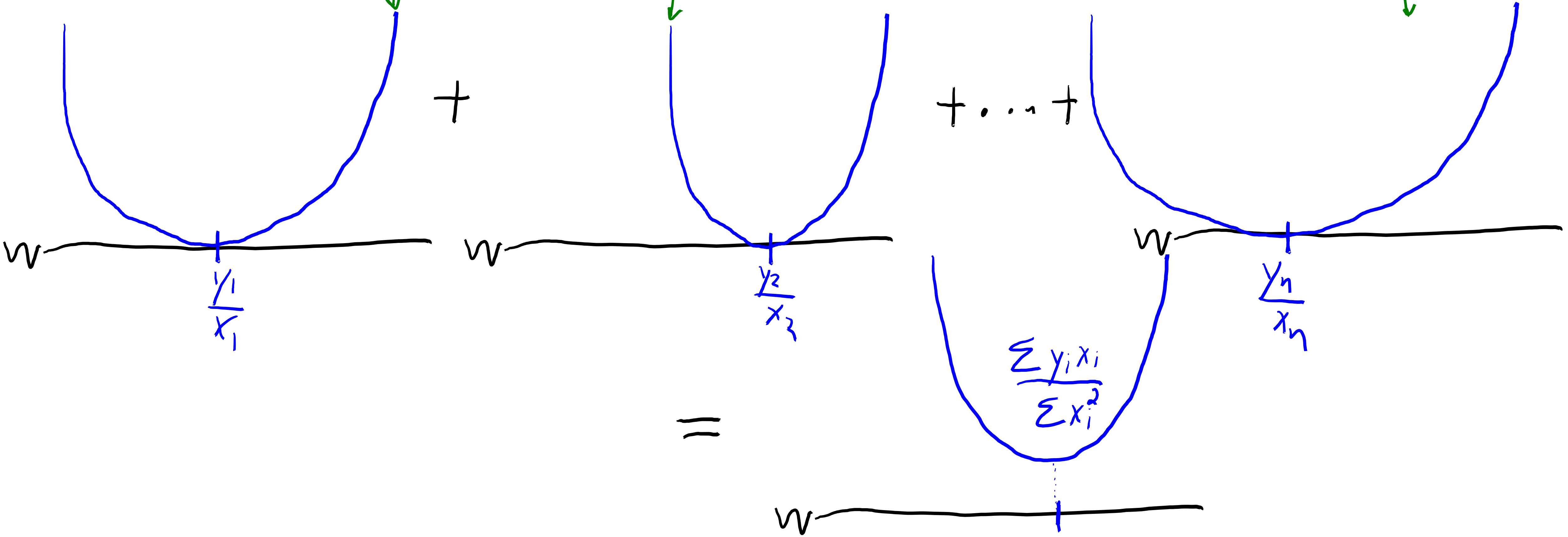
$$f''(w) = \sum_{i=1}^{n} x_i^2$$

  – Since $(\text{anything})^2$ is non-negative, $f''(w) \geq 0$ and this is a minimizer.

Least Squares Objective/Solution (Another View)

- Least squares minimizes a quadratic that is a sum of quadratics:

$$f(w) = (wx_1 - y_1)^2 + (wx_2 - y_2)^2 + (wx_3 - y_3)^2 + \cdots + (wx_n - y_n)^2$$



(pause)

Motivation: Combining Explanatory Variables

- Smoking is not the only contributor to lung cancer.
  - For example, there environmental factors like exposure to asbestos.
- How can we model the combined effect of smoking and asbestos?
- A simple way is with a 2-dimensional linear function:

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2}$$

Value of feature 2 in example 'i'

"weight" on feature 2.

"weight" of feature 1

Value of feature 1 in example 'i'

- We have a weight w_1 for feature '1' and w_2 for feature '2':

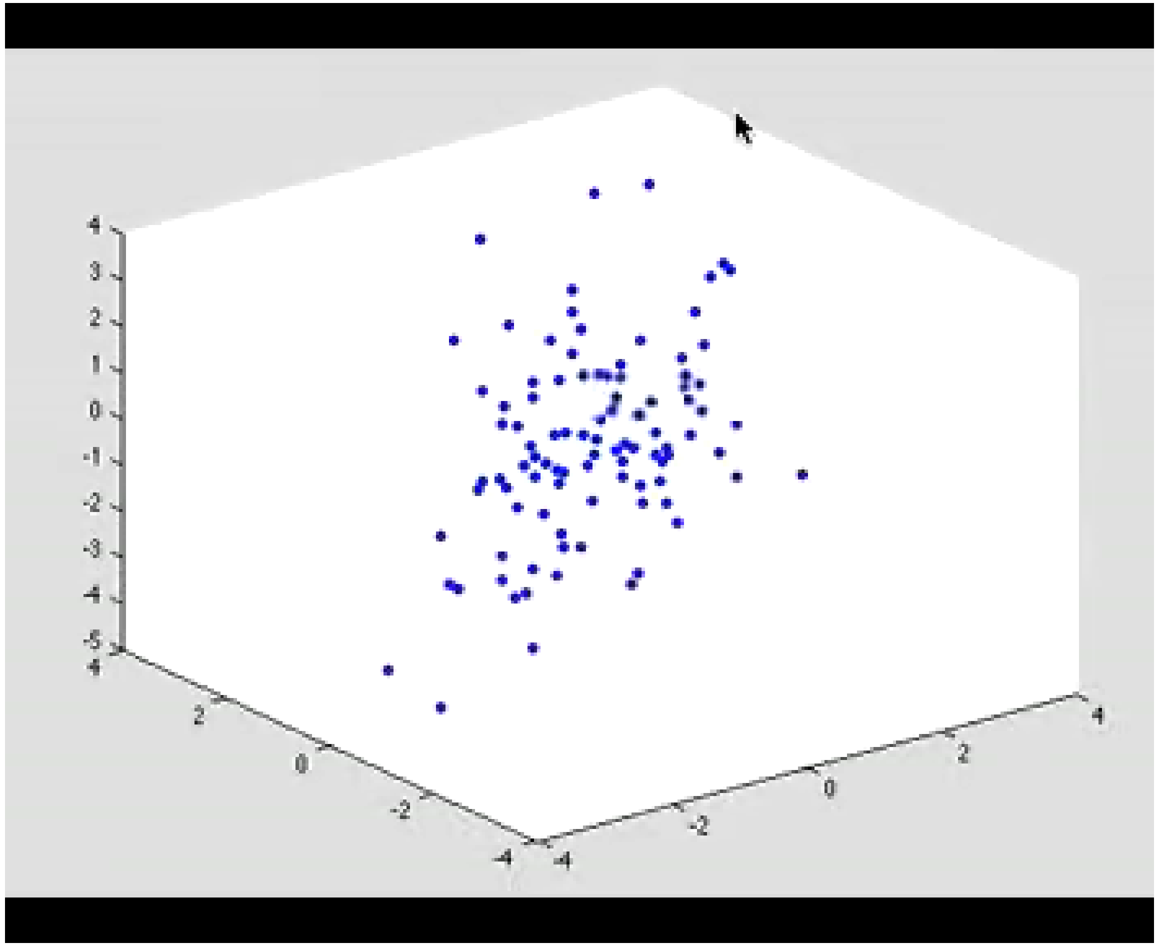
$$\hat{y}_i = 10(\#\text{cigarettes}) + 25(\#\text{asbetos})$$

Least Squares in 2-Dimensions

- Linear model:

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2}$$

- This defines a two-dimensional hyper-plane.



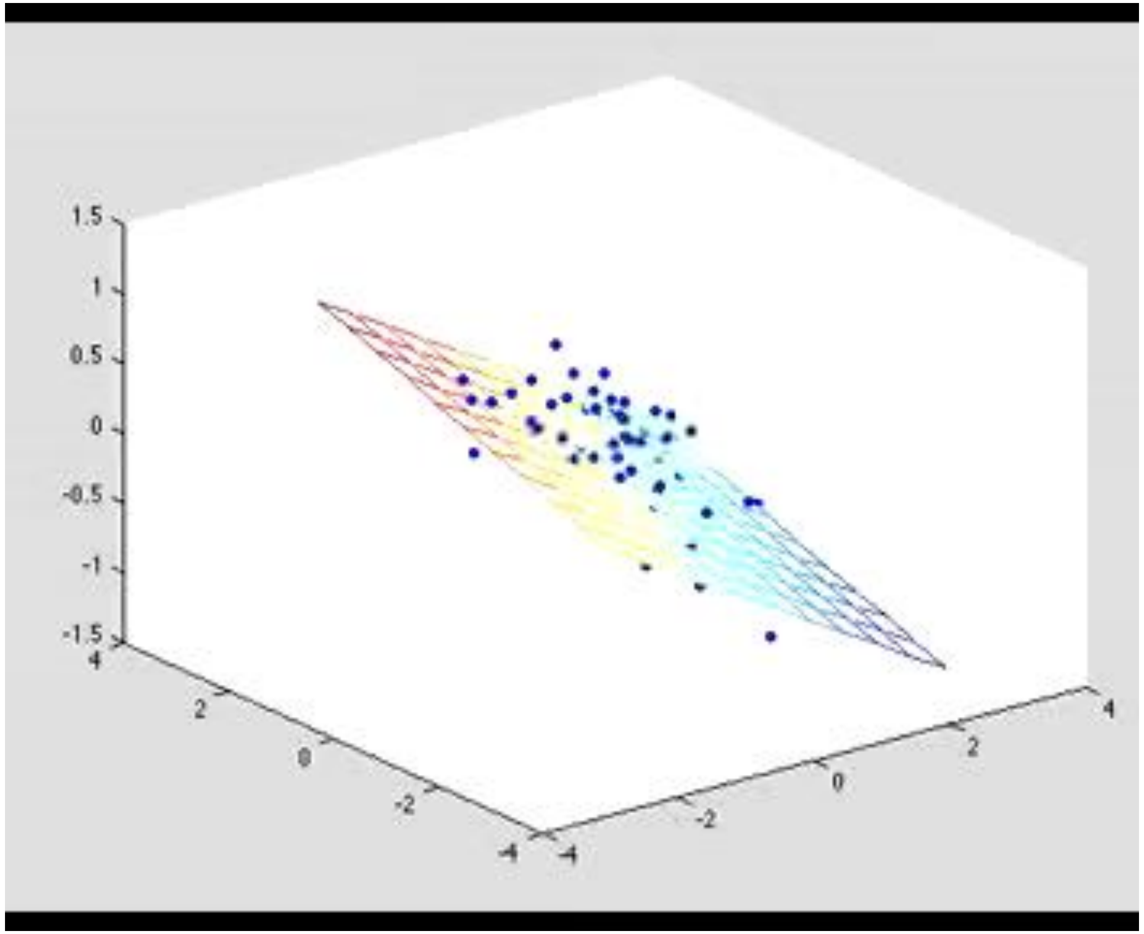
Least Squares in 2-Dimensions

- Linear model:

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2}$$

- This defines a two-dimensional hyper-plane.

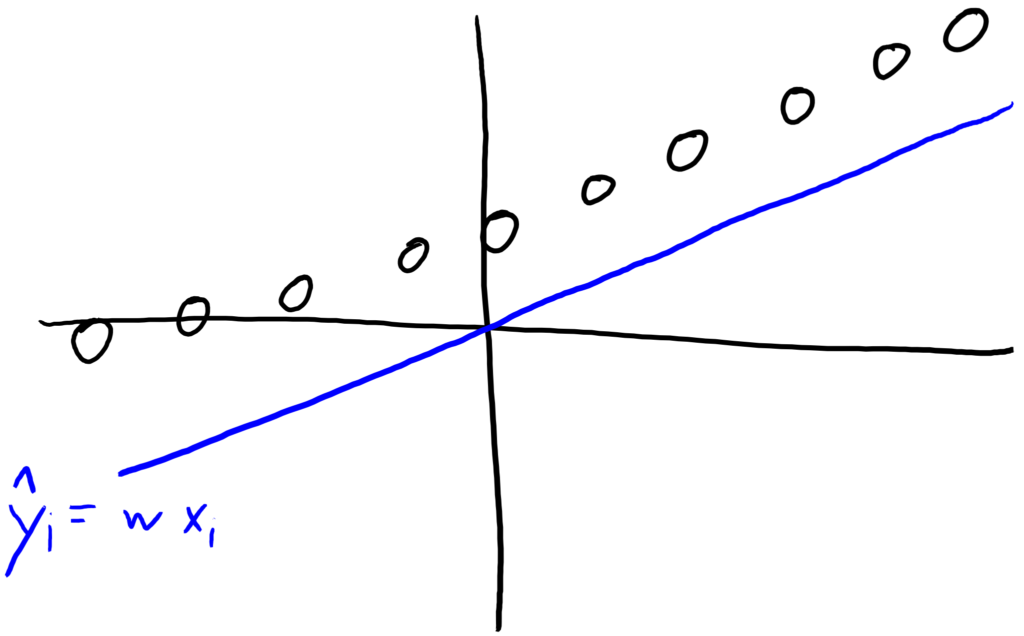
- Not just a line!



Why don’t we have a y-intercept?

– Linear model is $\hat{y}_i = wx_i$ instead of $\hat{y}_i = wx_i + w_0$ with y-intercept w_0.

– Without an intercept, if $x_i = 0$ then we must predict $\hat{y}_i = 0$.

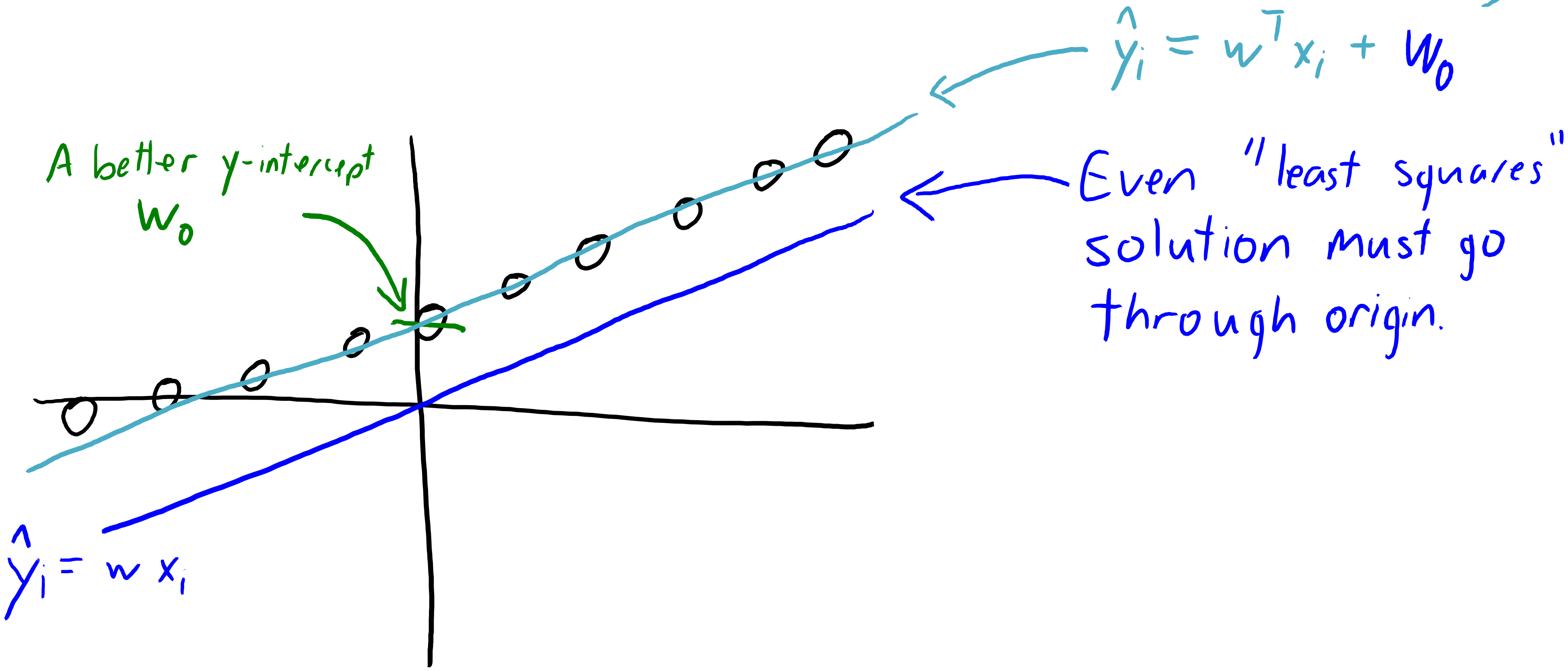


Even "least squares" solution must go through origin.

Why don't we have a y-intercept?

- Linear model is $\hat{y}_i = wx_i$ instead of $\hat{y}_i = wx_i + w_0$ with y-intercept w_0.
- Without an intercept, if $x_i = 0$ then we must predict $\hat{y}_i = 0$.

Adding y-intercept fixes this.



Adding a Bias Variable

- Simple trick to add a y-intercept ("bias") variable:
  - Make a new matrix "Z" with an extra feature that is always "1".

$$X = \begin{bmatrix} -0.1 \\ 0.3 \\ 0.2 \end{bmatrix} \qquad Z = \begin{bmatrix} \underbrace{\begin{matrix} 1 \\ 1 \\ 1 \end{matrix}}_{\text{"always 1"}} & \underbrace{\begin{matrix} -0.1 \\ 0.3 \\ 0.2 \end{matrix}}_{X} \end{bmatrix}$$

- Now use "Z" as your features in linear regression.
  - We'll use 'v' instead of 'w' as regression weights when we use features 'Z'.

$$\hat{y}_i = \underset{w_0}{\underset{\downarrow}{v_1}} \underset{1}{\underset{\downarrow}{z_{i1}}} + \underset{w_1}{\underset{\downarrow}{v_2}} \underset{x_{i1}}{\underset{\downarrow}{z_{i2}}} = w_0 + w_1 x_{i1}$$

- So we can have a non-zero y-intercept by changing features.
  - This means we can ignore the y-intercept in our derivations, which is cleaner.

Different Notations for Least Squares

- If we have 'd' features, the d-dimensional linear model is:

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2} + w_3 x_{i3} + \cdots + w_d x_{id}$$

  – In words, our model is that the output is a weighted sum of the inputs.

- We can re-write this in summation notation:

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

- We can also re-write this in vector notation:

$$\hat{y}_i = \underbrace{w^T x_i}_{\text{"inner product" between vectors}}$$

(assuming 'w' and x_i are column-vectors)

Notation Alert (again)

- In this course, all vectors are assumed to be column-vectors:

$$w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix} \qquad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \qquad x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix}$$

- So $w^T x_i$ is a scalar:

$$w^T x_i = \begin{bmatrix} w_1 & w_2 & \cdots & w_d \end{bmatrix} \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix} = w_1 x_{i1} + w_2 x_{i2} + \cdots + w_d x_{id}$$
$$= \sum_{j=1}^{d} w_j x_{id}$$

- So rows of 'X' are actually transpose of column-vector x_i:

$$X = \begin{bmatrix} \text{—} & x_1^T & \text{—} \\ \text{—} & x_2^T & \text{—} \\ & \vdots & \\ \text{—} & x_n^T & \text{—} \end{bmatrix}$$

Least Squares in d-Dimensions

- The linear least squares model in d-dimensions minimizes:

$$f(w) = \frac{1}{2}\sum_{i=1}^{n}(\underbrace{w^T x_i}_{\text{prediction}} - y_i)^2$$

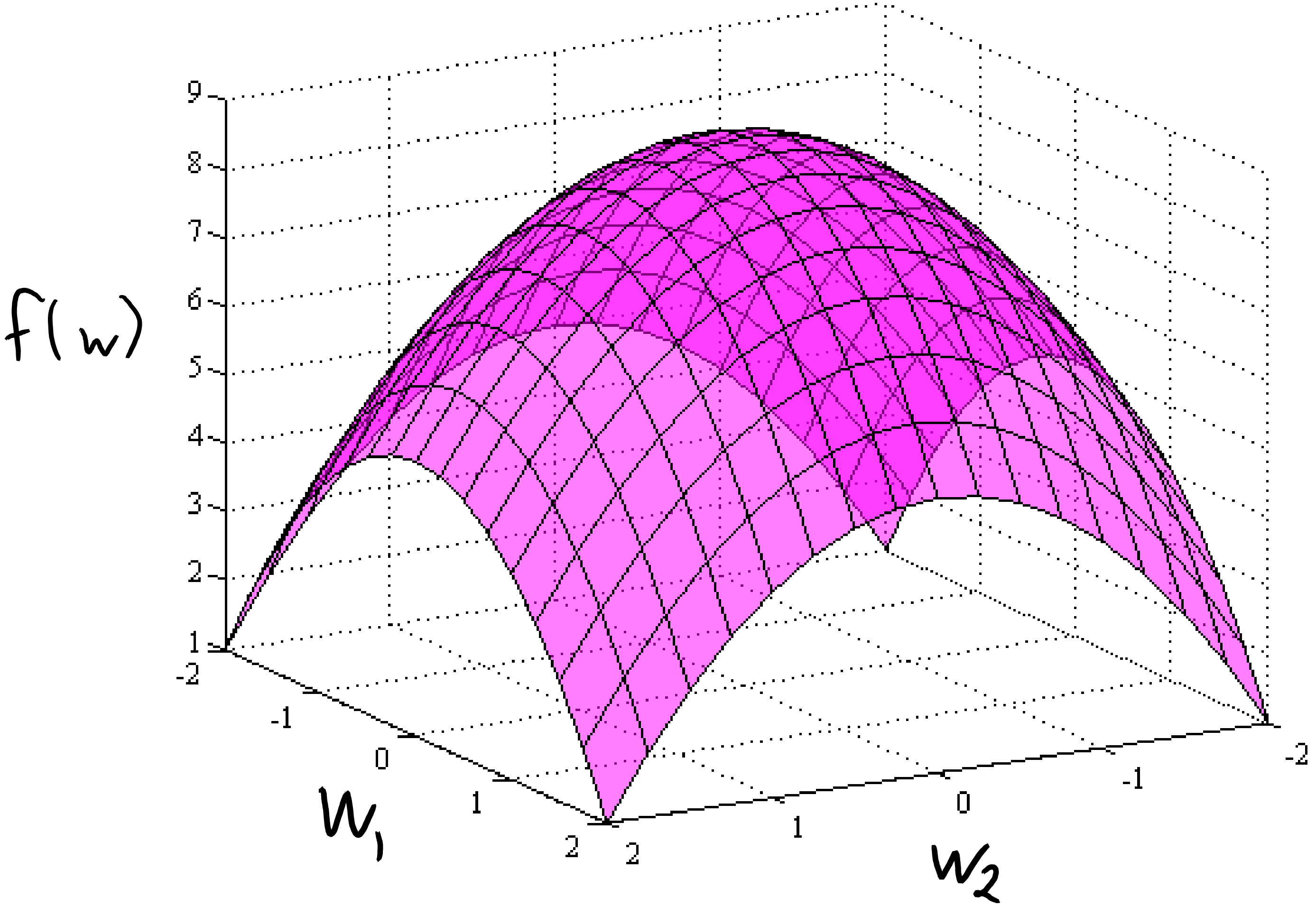
'w' is now a vector

prediction is inner product of 'w' and 'x_i' (linear combination of features)

"Error" is still the sum of squared differences between "true" y_i and our "prediction" $w^T x_i$

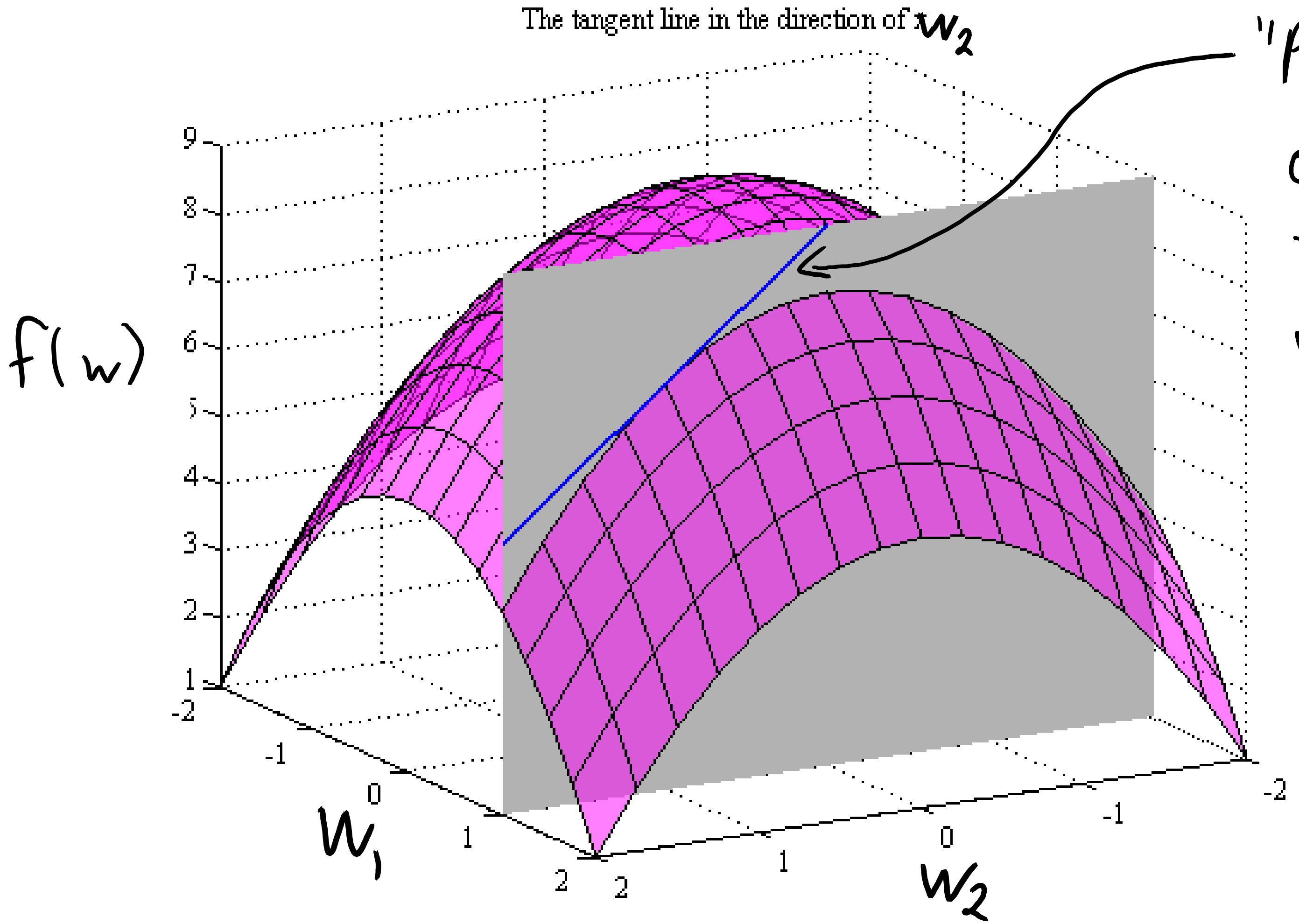
- Dates back to 1801: Gauss used it to predict location of Ceres.
- How do we find the best vector 'w' in 'd' dimensions?
  - Set the derivative of each variable ("partial derivative") to 0?

Partial Derivatives



http://msemac.redwoods.edu/~darnold/math50c/matlab/pderiv/index.xhtml

Partial Derivatives



"Partial" derivative of 'f' with respect to w_2 is the derivative with respect to w when all other variables are held fixed.

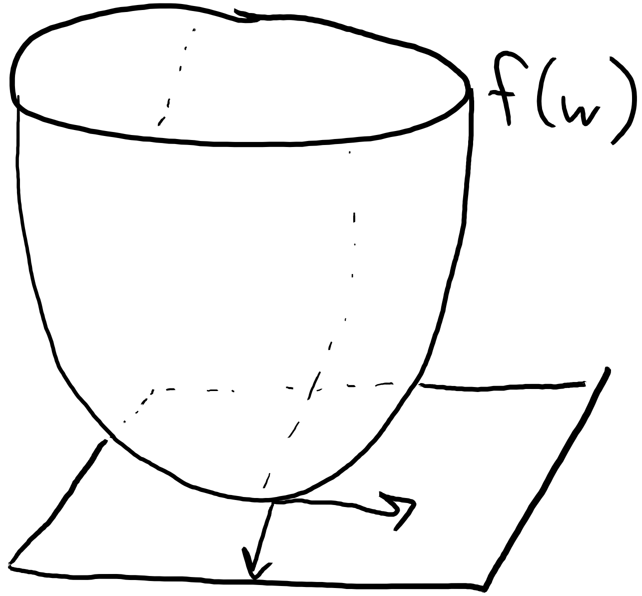
Denoted by $\frac{\partial}{\partial w_2}$ for variable w_2

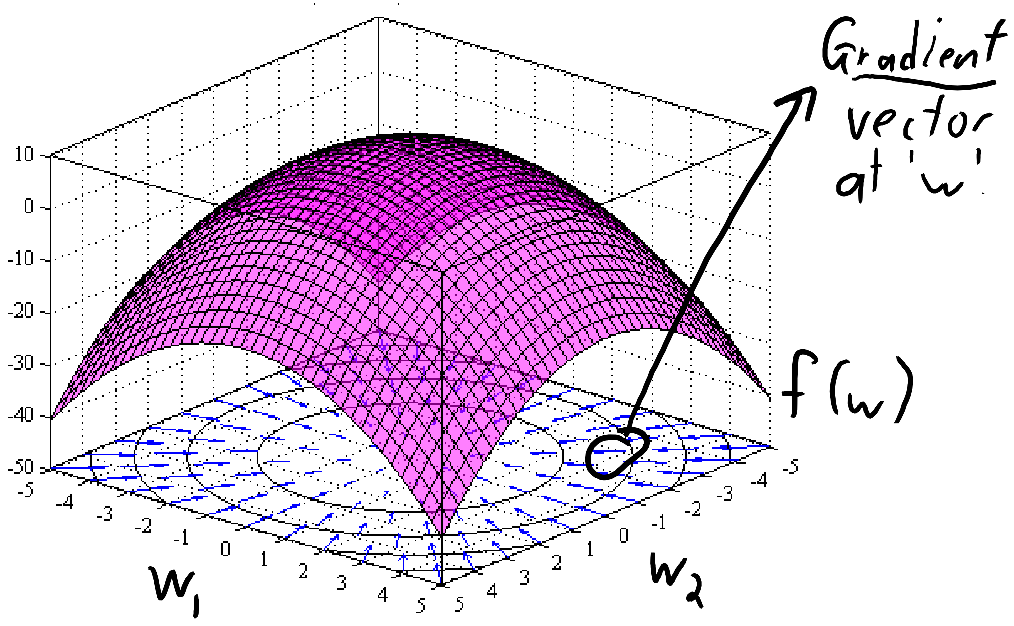
http://msemac.redwoods.edu/~darnold/math50c/matlab/pderiv/index.xhtml

Gradient and Critical Points in d-Dimensions

- Generalizing “set the derivative to 0 and solve” in d-dimensions:
  - Find ‘w’ where the gradient vector equals the zero vector.
- Gradient is vector with partial derivative ‘j’ in position ‘j’:

$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix}$$





Tangent slope is 0 in every direction at minimizer.

ac.redwoods.edu/~darnold/math50c/matlab/pderiv/index.xhtml

Least Squares Partial Derivatives (1 Example)

- The linear least squares model in d-dimensions for 1 example:

$$f(w_1, w_2, \dots, w_d) = \frac{1}{2}(\hat{y}_i - y_i)^2 = \frac{1}{2}\hat{y}_i^2 - \hat{y}_i y_i + \frac{1}{2} y_i^2$$

$$\hat{y}_i = w_1 x_{i1} + w_2 x_{i2} + \dots w_d x_{id}$$

$$= \frac{1}{2}\left(\sum_{j=1}^{d} w_j x_{ij}\right)^2 + \left(\sum_{j=1}^{d} w_j x_{ij}\right) y_i + \frac{1}{2} y_i^2$$

- Computing the partial derivative for variable '1':

$$\frac{\partial}{\partial w_1} f(w_1, w_2, \dots, w_d) = \left(\sum_{j=1}^{d} w_j x_{ij}\right) x_{i1} - y_i x_{i1} + 0$$

$$= \left(\sum_{j=1}^{d} w_j x_{ij} - y_i\right) x_{i1}$$

$$= (w^T x_i - y_i) x_{i1}$$

Least Squares Partial Derivatives (‘n’ Examples)

- Linear least squares partial derivative for variable 1 on example ‘i’:

$$\frac{\partial}{\partial w_1} f(w_1, w_2, \dots, w_d) = (w^T x_i - y_i) x_{i1}$$

- For a generic variable ‘j’ we would have:

$$\frac{\partial}{\partial w_j} f(w_1, w_2, \dots, w_d) = (w^T x_i - y_i) x_{ij}$$

- And if ‘f’ is summed over all ‘n’ examples we would have:

$$\frac{\partial}{\partial w_j} f(w_1, w_2, \dots, w_d) = \sum_{i=1}^{n} (w^T x_i - y_i) x_{ij}$$

- Unfortunately, the partial derivative for w_j depends on all $\{w_1, w_2, \dots, w_d\}$
  - I can’t just “set equal to 0 and solve for w_j”.

Gradient and Critical Points in d-Dimensions

- Generalizing "set the derivative to 0 and solve" in d-dimensions:
  – Find 'w' where the gradient vector equals the zero vector.
- Gradient is vector with partial derivative 'j' in position 'j':

$$\nabla f(w) = \begin{bmatrix} \frac{\partial f}{\partial w_1} \\ \frac{\partial f}{\partial w_2} \\ \vdots \\ \frac{\partial f}{\partial w_d} \end{bmatrix}$$

For linear least squares:

$$\nabla f(w) = \begin{bmatrix} \sum_{i=1}^{n} (w^T x_i - y_i) x_{i1} \\ \sum_{i=1}^{n} (w^T x_i - y_i) x_{i2} \\ \vdots \\ \sum_{i=1}^{n} (w^T x_i - y_i) x_{id} \end{bmatrix}$$

Claims for linear least square:

1. Finding a 'w' where $\nabla f(w) = 0$ can be done by solving a system of linear equations.
2. All 'w' where $\nabla f(w) = 0$ are minimizers.

Matrix/Norm Notation (MEMORIZE/STUDY THIS)

- To solve the d-dimensional least squares, we use matrix notation:
  - We use 'y' as an "n times 1" vector containing target 'y_i' in position 'i'.
  - We use 'x_i' as a "d times 1" vector containing features 'j' of example 'i'.
    - We're now going to be careful to make sure these are column vectors.
  - So 'X' is a matrix with the x_i^T in row 'i'.

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \qquad x_i = \begin{bmatrix} x_{i1} \\ x_{i2} \\ \vdots \\ x_{id} \end{bmatrix} \qquad X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & \cdots & x_{2d} \\ \vdots & & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} \end{bmatrix} = \begin{bmatrix} \text{---} & x_1^T & \text{---} \\ \text{---} & x_2^T & \text{---} \\ & \vdots & \\ \text{---} & x_n^T & \text{---} \end{bmatrix}$$

Matrix/Norm Notation (MEMORIZE/STUDY THIS)

- To solve the d-dimensional least squares, we use matrix notation:
  - Our prediction for example 'i' is given by the scalar $w^T x_i$.
  - Our predictions for all 'i' (n times 1 vector) is the matrix-vector product Xw.

$$\hat{y}_i = w^T x_i$$

$$\hat{y} = Xw = \underbrace{\begin{bmatrix} — & x_1^T & — \\ — & x_2^T & — \\ & \vdots & \\ — & x_n^T & — \end{bmatrix}}_{X} \underbrace{\begin{bmatrix} | \\ w \\ | \end{bmatrix}}_{w} = \begin{bmatrix} x_1^T w \\ x_2^T w \\ \vdots \\ x_n^T w \end{bmatrix}$$

Prediction for example 'i' is in row 'i'

Also, because $w^T x_i$ is a scalar, we have $w^T x_i = x_i^T w$.

(e.g., $[5]^T = [5]$)

Matrix/Norm Notation (MEMORIZE/STUDY THIS)

- To solve the d-dimensional least squares, we use matrix notation:
  - Our prediction for example 'i' is given by the scalar w^Tx_i.
  - Our predictions for all 'i' (n times 1 vector) is the matrix-vector product Xw.
  - Residual vector 'r' gives difference between prediction and y_i (n times 1).
  - Least squares can be written as the squared L2-norm of the residual.

$$f(w) = \sum_{i=1}^{n} (\underbrace{w^Tx_i - y_i}_{r_i})^2 = \sum_{i=1}^{n} (r_i)^2$$

$$= \sum_{i=1}^{n} r_i r_i$$

$$= r^Tr$$

$$= \|r\|^2 = \|Xw - y\|^2$$

$$r = \hat{y} - y = \underbrace{Xw}_{\hat{y}} - y = \begin{bmatrix} w^Tx_1 \\ w^Tx_2 \\ \vdots \\ w^Tx_n \end{bmatrix} - \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} w^Tx_1 - y_1 \\ w^Tx_2 - y_2 \\ \vdots \\ w^Tx_n - y_n \end{bmatrix}$$

r_2 is difference for example 2.

Summary

- Regression considers the case of a numerical y_i.
- Least squares is a classic method for fitting linear models.
  - With 1 feature, it has a simple closed-form solution.
  - Can be generalized to 'd' features.
- Gradient is vector containing partial derivatives of all variables.
- Matrix notation for expressing least squares problem.
- Next time: minimizing $\frac{1}{2}\sum_{i=1}^{n}(w^Tx_i - y_i)^2$ in terms of 'w' is:

$$w = (X'X)\backslash(X'y)$$

(in Julia)