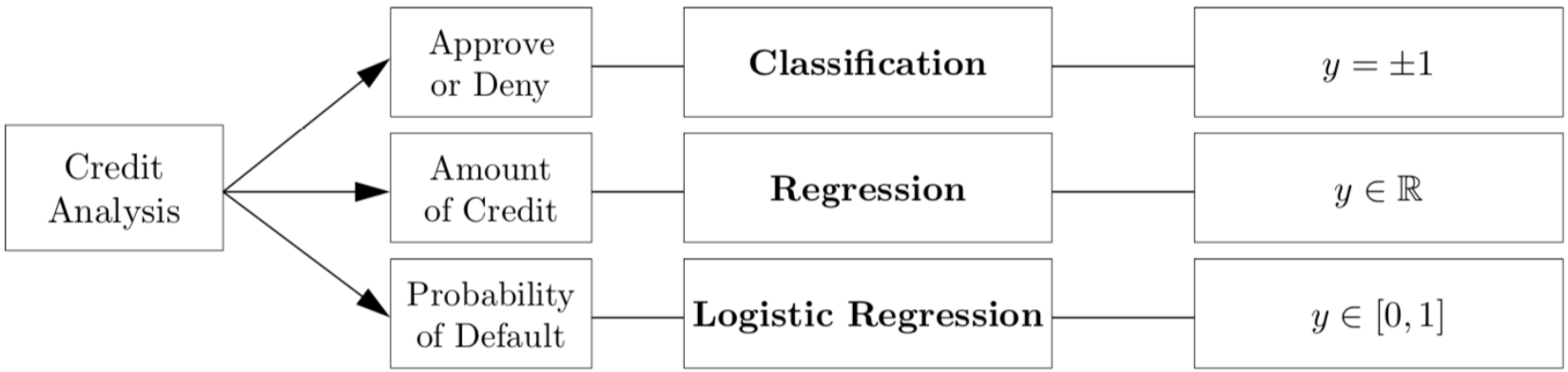
Linear Regression

Georgia Tech

The Linear Model

So far we've dealt with classification, where the target function maps feature vectors to discrete classes, but the linear model is more versatile. Consider the credit analysis problem:



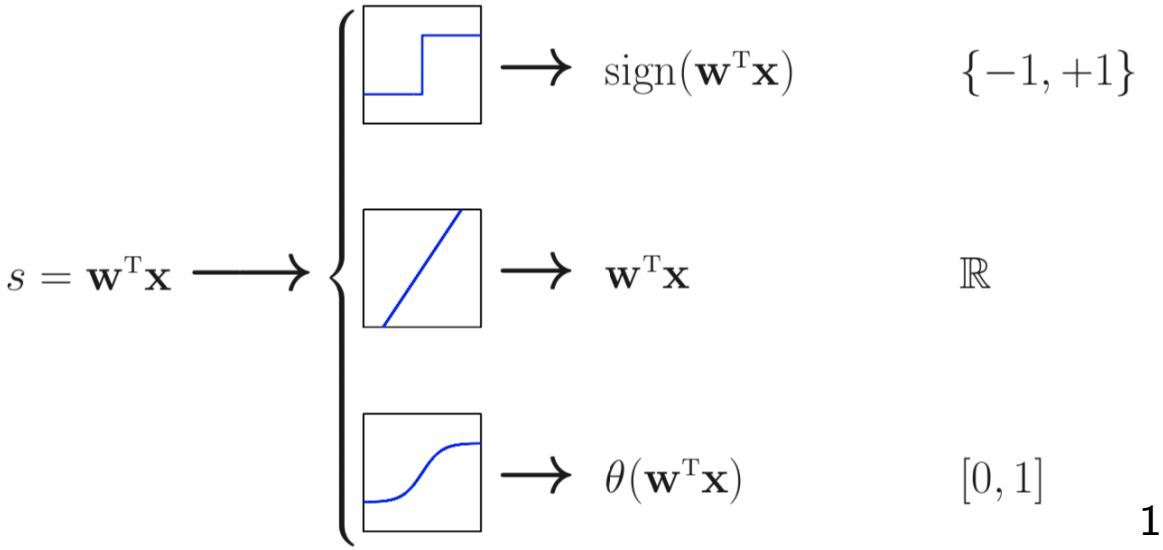
We can use the linear model to learn

- a yes/no (perceptron)
- an arbitrary real number (linear regression)
- a probability (logistic regression)

As we'll see later, we can even learn to separate classes that are not linearly separable due to their nature, not noise in the data set.

Georgia Tech

The Linear Signal



[1]

$$y = \theta(s)$$

Three different error measures (loss functions):

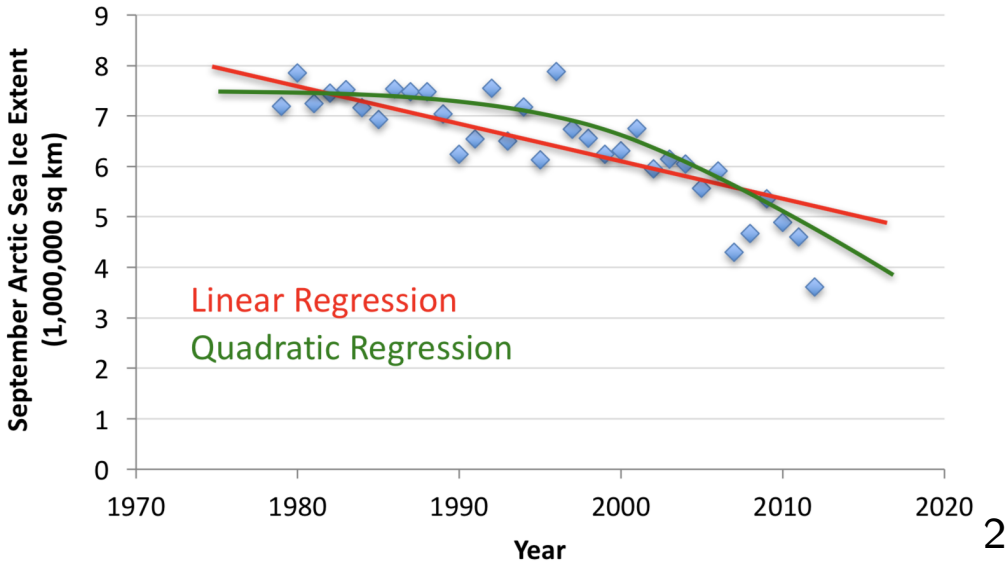
- Perceptron: Classification error (0-1 loss)
- Linear Regression: Mean square error
- Logistic Regression: Cross-entropy error

The different error measures lead to different algorithms for minimizing the error.

[1] http://www.cs.rpi.edu/~magdon/courses/learn/slides.html

Linear Regression

In linear regression the target function maps feature vectors in $\mathbb{R}^{d+1}$ to arbitrary real values.



[2]

- In linear binary classification we assume that there is a line that separates classes acceptably well.
- In linear regression we assume that there is a line that fits the data acceptably well.

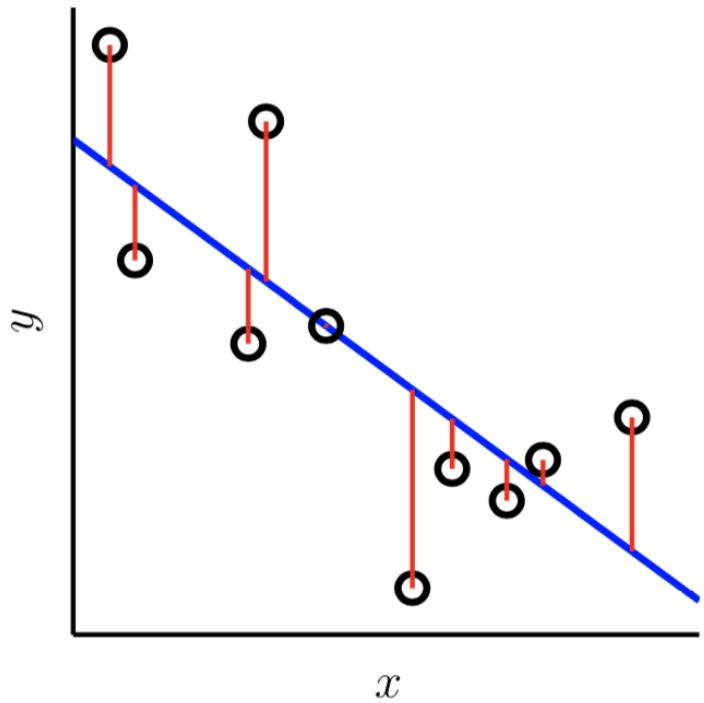
[2] https://www.cc.gatech.edu/~bboots3/CS4641-Fall2018/Lecture3/03_LinearRegression.pdf

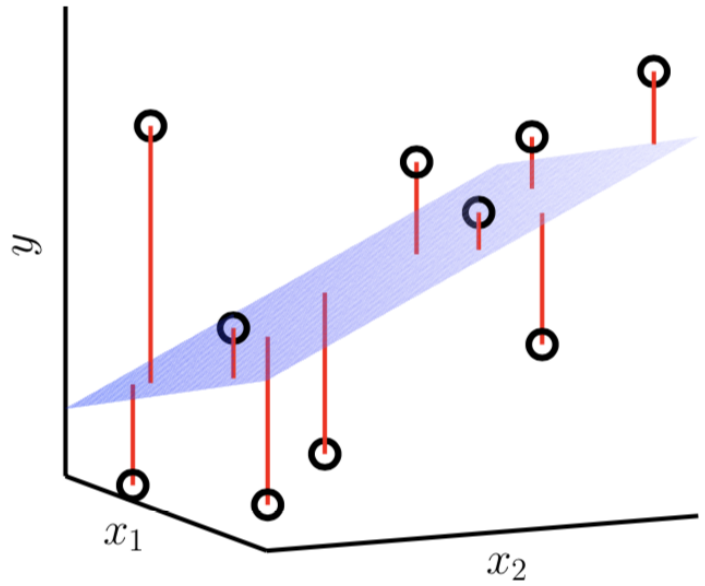
Georgia Tech

Error for Linear Regression

In linear regression we minimize the mean square error (MSE) between $h(\vec{x})$ and y.

$$E_{in} = \frac{1}{N}\sum_{n=1}^{N}(h(\vec{x_n} - y_n)^2$$





Matrix Representation of $E_{in}(\vec{w})$

We can represent the problem in matrix form:

$$\underbrace{\mathrm{X} = \begin{bmatrix} \text{—}\mathbf{x}_1\text{—} \\ \text{—}\mathbf{x}_2\text{—} \\ \vdots \\ \text{—}\mathbf{x}_N\text{—} \end{bmatrix}}_{\text{data matrix, } N \times (d+1)} \qquad \underbrace{\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}} \qquad \underbrace{\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} \mathbf{w}^{\mathrm{T}}\mathbf{x}_1 \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_2 \\ \vdots \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_N \end{bmatrix} = \mathrm{X}\mathbf{w}}_{\text{in-sample predictions}}$$

Then:

$$\begin{aligned} E_{\text{in}}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N}(\hat{y}_n - y_n)^2 \\ &= \tfrac{1}{N}\| \hat{\mathbf{y}} - \mathbf{y} \|_2^2 \\ &= \tfrac{1}{N}\| \mathrm{X}\mathbf{w} - \mathbf{y} \|_2^2 \\ &= \tfrac{1}{N}(\mathbf{w}^{\mathrm{T}}\mathrm{X}^{\mathrm{T}}\mathrm{X}\mathbf{w} - 2\mathbf{w}^{\mathrm{T}}\mathrm{X}^{\mathrm{T}}\mathbf{y} + \mathbf{y}^{\mathrm{T}}\mathbf{y}) \end{aligned}$$

Georgia Tech

Minimizing $E_{in}(\vec{w})$

For linear regression, h is a linear combination of the components of $\vec{x}$:

$$h(x) = \vec{w}^T \vec{x}$$

And minimizing the error is expressed as the optimization problem:

$$w_{lin} = \underset{\vec{w} \in \mathbb{R}^{d+1}}{\operatorname{argmin}} E_{in}(\vec{w})$$

Georgia Tech

Minimizing Matrix Repreeantation of $E_{in}(\vec{w})$

Since $E_{in}(\vec{w})$ is differentiable we can set the gradient to 0:

$$\nabla E_{in}(\vec{w}) = \vec{0}$$

The gradient of our matrix representation of $E_{in}(\vec{w})$ is

$$\nabla E_{in}(\vec{w}) = \frac{2}{N}(X^T X\vec{w} - X^T\vec{y})$$

which is $\vec{0}$ when

$$X^T X\vec{w} = X^T\vec{y}$$

If we assume $X^T X$ is invertible, then $\vec{w} = X^\dagger\vec{y}$, leading to the one-step algorithm for linear regression . . .

Georgia Tech

Linear Regression Algorithm

Linear Regression Algorithm:

1. Construct the matrix X and the vector $\mathbf{y}$ from the data set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$, where each $\mathbf{x}$ includes the $x_0 = 1$ coordinate,

$$\underbrace{\mathrm{X} = \begin{bmatrix} —\mathbf{x}_1— \\ —\mathbf{x}_2— \\ \vdots \\ —\mathbf{x}_N— \end{bmatrix}}_{\text{data matrix}}, \qquad \underbrace{\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}.$$

2. Compute the pseudo inverse $\mathrm{X}^\dagger$ of the matrix X. If $\mathrm{X}^\mathrm{T}\mathrm{X}$ is invertible,

$$\mathrm{X}^\dagger = (\mathrm{X}^\mathrm{T}\mathrm{X})^{-1}\mathrm{X}^\mathrm{T}$$

3. Return $\mathbf{w}_{\text{lin}} = \mathrm{X}^\dagger\mathbf{y}$.

Georgia Tech

Closing Thoughts

- The linear regression algorithm demonstrates some common themes in machine learning:
  - manipulate the error function into a form that allows us to use to mathematical tricks to simplify the problem
  - make simplifying assumptions that make the theory clean but typically work well in practice
- Linear regression is well-studied in statistics
- Some wouldn't consider linear regression to be machine learning becuase it has an analytic rather than algorithmic solution
  - important ideas in linear regression appear in other algorithms (e.g., minimizing gradient)
  - linear regression is an important tool in a data scientist's tool box

Georgia Tech