CSE 6363 - Machine Learning

Linear Regression

Alex Dillhoff

University of Texas at Arlington

Linear Models

We begin this course with a simple task:

Fit a linear model to some given data.

Linear Models

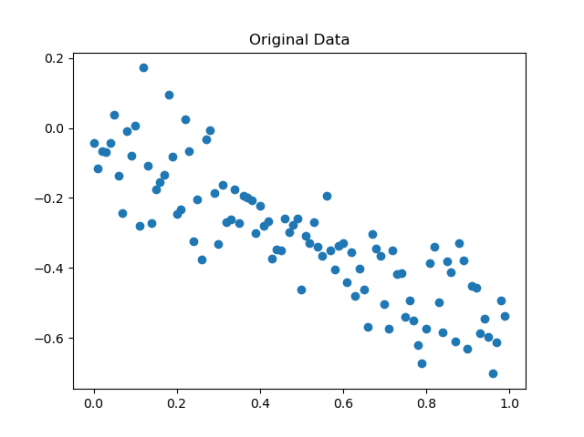

Figure 1: Data generated from a line with Gaussian noise.

Linear Models

We can make up plenty of stories about what this data represents:

- Age vs relative wealth
- Time practicing vs ELO rating
- Housing prices versus size
- ...

Linear Models

The x values of the data are the raw input **features**.

The y values represent the **observation** or **target**.

Linear Models

If we fit a model to the data, we can make predictions about our dataset as well as novel data points.

This is the **inference** task.

Linear Models

Since our dataset looks like it could be generalized with a line, let's see if we can fit some line to it.

$$y = mx + b$$

Linear Models

The slope-intercept form has 2 parameters:

- m - The slope of the line
- b - The bias term

Linear Models

We'll change the notation of these parameters such that they fit in a parameter vector:

$$\mathbf{w} = (w_0, w_1)$$

Linear Models

In general, we can represent our input features with a vector $\mathbf{x} \in \mathbb{R}^d$, where d is the number of features.

For our simple dataset, $d = 1$.

Linear Models

We will also represent our model as a function $h(\mathbf{x}; \mathbf{w})$ such that

$$h(\mathbf{x}; \mathbf{w}) = w_0 + w_1 x_1$$

Linear Models

Where is x_0?

The only parameter that depends on the input features $\mathbf{x}$ is w_1.

For convenience, we will prepend a constant $x_0 = 1$ to $\mathbf{x}$ such that

$$\mathbf{x} = (x_0, x_1)$$

Linear Models

Now the model can be written more compactly as

$$h(\mathbf{x}; \mathbf{w}) = \sum_{i=0}^{d} w_i x_i = \mathbf{w}^T \mathbf{x}$$

Linear Models

Given this representation, we can manually tune the parameters $\mathbf{w}$ until we can visually inspect a ”good” line.

After some time, let’s say we land on $w_0 = 0$ and $w_1 = -0.7$.

Linear Models

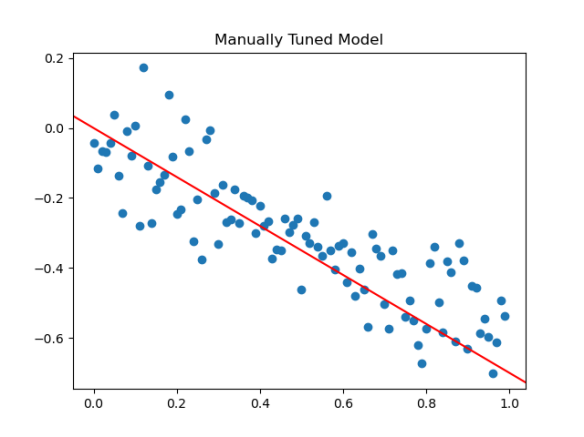

Figure 2: Fitting our model with manual tuning. $\mathbf{w} = (0, -0.7)$

Determining Fitness

Those choices may look good qualitatively, but we need a way to determine how good they are **quantitatively**.

The classic choice here is to average the squared error between each of the true observations and our model's prediction.

Determining Fitness

For this, we introduce a **cost function** or **loss function**:

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{n} (h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i)^2.$$

Determining Fitness

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{n}(h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i)^2$$

There are several loss functions that we could try (and we'll explore more later in the semester).

However, no matter what $\mathbf{w}$ we choose for this model, we can never achieve an error of 0.

Determining Fitness

With our loss function defined, we can define the task of finding the most optimal choice of $\mathbf{w}$ with respect to the loss function as

$$\min_{\mathbf{w}} J(\mathbf{w})$$

Stochastic Gradient Descent

Before looking at an analytical solution to this problem, we turn to **stochastic gradient descent**.

1. Initialize **w** to some values (random or heuristic-based)
2. Evaluate the current model performance
3. Update the parameters based on the previous step
4. Repeat steps 2 and 3 until convergence

Stochastic Gradient Descent

How do we modify w such that $J(\mathbf{w})$ will decrease?

$\nabla J(\mathbf{w})$ yields the direction of greatest descent.

Stochastic Gradient Descent

How do we know when we've converged to a solution?

For this particular problem, the global minima is the only minima:

$$\nabla J(\mathbf{w}) = 0.$$

Stochastic Gradient Descent

How do we know when we've converged to a solution?

This is not the case for models that are not convex.

Stochastic Gradient Descent

Our choice of loss function is convenient because the gradient is simple to calculate:

$$\frac{d}{d\mathbf{w}} J(\mathbf{w}) = (h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i)\mathbf{x}_i$$

Stochastic Gradient Descent

Given the gradient, how do we update our current parameters?

The **update rule**:

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \sum_{i=1}^{n} (h(\mathbf{x}_i; \mathbf{w}_t) - \mathbf{y}_i)\mathbf{x}_i.$$

Stochastic Gradient Descent

The hyperparameter α controls the size of the step during the gradient update.

If the value is too large, the model may never converge to a solution.

If the value is too small, it is more likely to get stuck in local minima.

Stochastic Gradient Descent

Consider a much simpler function like $f(w) = w^2$.

The update step of this function is

$$w_{t+1} = w_t - \alpha 2w$$

Stochastic Gradient Descent

If our current parameter $w = 1$ and we update with $\alpha = 1$, the resulting weight is

$$\begin{aligned} w_{t+1} &= 1 - (1) * 2(1) \\ &= -1 \end{aligned}$$

That results in a huge step that only puts our estimate on the other side of the gradient landscape.

Stochastic Gradient Descent

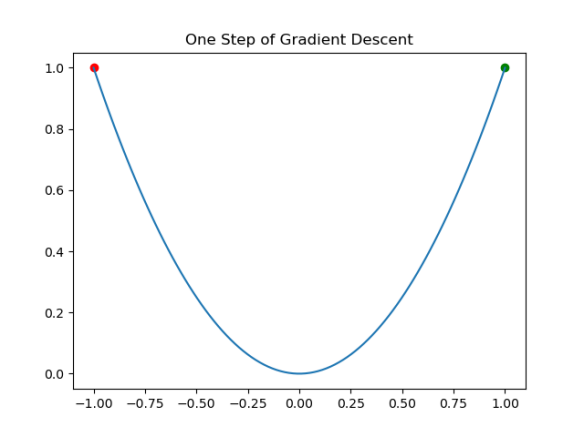

Figure 3: One step of SGD with function $f(w) = w^2$. The green point is the parameter estimate before the update.

Stochastic Gradient Descent

Another benefit to this rule is that it allows for **batch updates**.

When we work with very large datasets, it will be impossible to update the gradient based on the entire set.

A Probabilistic Approach

We can also take a probabilistic approach and show that minimizing the sum of squares is equivalent to maximizing the posterior probability of our model.

A Probabilistic Approach

Recall Bayes' Theorem:

$$p(\mathbf{w}|\mathbf{X}) = \frac{p(\mathbf{X}|\mathbf{w})p(\mathbf{w})}{p(\mathbf{X})}$$

A Probabilistic Approach

Let's adapt this to our model, where $\mathbf{X} \in \mathbb{R}^{n \times d}$ are the features and $\mathbf{Y} \in \mathbb{R}^n$ are the observations.

$$p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{w})p(\mathbf{w}|\mathbf{X})}{p(\mathbf{Y}|\mathbf{X})}$$

A Probabilistic Approach

The choice of using least squares as our loss function also has statistical motivations.

We start with a reasonable assumption of a relationship between the features of the data and the observed output.

$$\hat{\mathsf{Y}} = f(\mathsf{X}) + \epsilon$$

A Probabilistic Approach

ϵ is an error term that is independent of the features and has a mean of 0.

It represents

- sampling error,
- random noise,
- and any effects that are not captured from our model f.

A Probabilistic Approach

Given our model $h(\mathbf{x}_i; \mathbf{w})$, we can define *epsilon* as

$$\epsilon_i = h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i.$$

A Probabilistic Approach

Another reasonable assumption: the discrepencies ϵ_i are i.i.d. with variance σ^2 and Gaussian PDF f.

Thus, the likelihood is

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) = \prod_{i=1}^{n} f(\epsilon_i; \sigma).$$

A Probabilistic Approach

Recalling the definition of a Gaussian PDF:

$$f(\epsilon_i; \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{\epsilon_i^2}{2\sigma^2} \right).$$

A Probabilistic Approach

With the new parameter σ, the posterior distribution changes slightly:

$$p(\mathbf{w}|\mathbf{X}, \mathbf{Y}, \sigma) = \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma)p(\mathbf{w}|\mathbf{X}, \sigma)}{p(\mathbf{Y}|\mathbf{X}, \sigma)}.$$

A Probabilistic Approach

The prior term $p(\mathbf{Y}|\mathbf{X}, \sigma)$ is a normalizing constant which ensures that the posterior is a valid probability distribution.

A Probabilistic Approach

One last assumption...

Assuming that all values for $\mathbf{w}$ are equally likely, $p(\mathbf{w}|\mathbf{X}, \sigma)$ becomes constant.

A Probabilistic Approach

If both $p(\mathbf{w}|\mathbf{X}, \sigma)$ and $p(\mathbf{Y}|\mathbf{X}, \sigma)$ are treated as constants, then we only need to worry about the likelihood function $p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma)$.

A Probabilistic Approach

Examining the likelihood function from this perspective reveals the connection to our previous solution for determining optimal $\mathbf{w}$:

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) = -\frac{n}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i)^2 \right).$$

A Probabilistic Approach

Therefore, maximizing the likelihood $p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma)$ is equivalent to minimizing the sum of squares loss function.

In practice, we would use the negative log of the likelihood function since it is monotonically decreasing.

The Normal Equations

You may recall another approach to least squares solutions when studying Linear Algebra.

Recall the **normal equations** $A^TA\mathbf{x} = A^T\mathbf{b}$.

The Normal Equations

These are derived by first projecting the observed data points $\mathbf{b}$ on to the column space of A and solving

$$A\mathbf{x} = \hat{\mathbf{b}},$$

where $\hat{\mathbf{b}} = \text{proj}_{\text{Col}A}\mathbf{b}$.

The Normal Equations

The vector $\mathbf{b} - \hat{\mathbf{b}}$ is orthogonal to Col$A$, so the product

$$A^T(\mathbf{b} - \hat{\mathbf{b}})$$

should be 0.

The Normal Equations

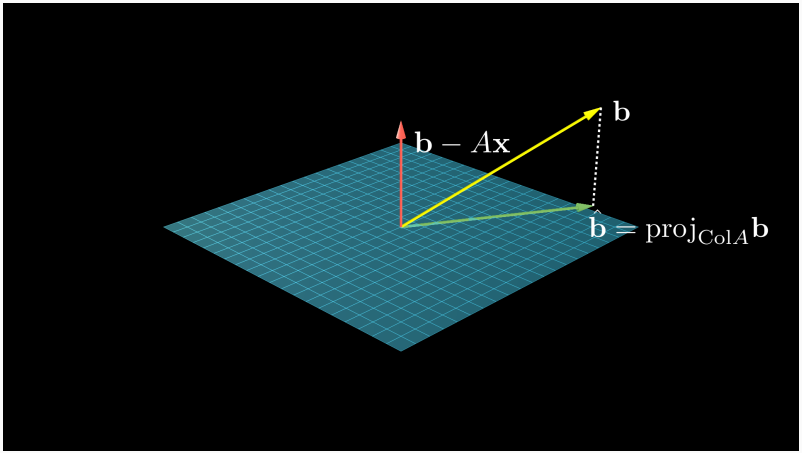

Figure 4: The projection of $\mathbf{b}$ onto Col$A$.

The Normal Equations

Rewriting this yields

$$A^T(\mathbf{b} - A\mathbf{x}) = \mathbf{0}$$
$$A^T\mathbf{b} - A^TA\mathbf{x} = \mathbf{0}.$$

Thus, each least-squares solution of $A\mathbf{x} = \mathbf{b}$ satisfies

$$A^TA\mathbf{x} = A^T\mathbf{b}.$$

The Normal Equations

To adapt this with our problem, we'll need to change the notation a bit:

$$\mathbf{X}^T\mathbf{X}\beta = \mathbf{X}^T\mathbf{y}$$

The Normal Equations

The design matrix $\mathbf{X}$ represents the the features of our data points:

$$\mathbf{X} = \begin{bmatrix} x_0^{(0)} & x_1^{(0)} \\ x_0^{(1)} & x_1^{(1)} \\ \vdots & \vdots \\ x_0^{(n)} & x_1^{(n)} \end{bmatrix}.$$

The Normal Equations

We are solving for our parameter vector

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}.$$

The Normal Equations

Finally, we solve for the parameter vector β:

$$\beta = (\mathbf{X}^T\mathbf{X})^{-1}X^T\mathbf{y}.$$

The Normal Equations

There is way to derive the normal equations from a probabilistic perspective starting with our likelihood function

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) = -\frac{n}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2} \sum_{i=1}^{n} (h(\mathbf{x}_i; \mathbf{w}) - \mathbf{y}_i)^2 \right).$$

The Normal Equations

Taking the natural log of this function yields

$$-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(h(\mathbf{x}_i;\mathbf{w})-\mathbf{y}_i)^2-\frac{n}{2}\ln(\sigma^2)-\frac{n}{2}\ln(2\pi).$$

The Normal Equations

We have established that maximizing the likelihood is equivalent to minimizing the sum-of-squares error.

Thus, we can determine the optimal parameters by finding the critical point of the likelihood function with respect to $\mathbf{w}$.

The Normal Equations

First, take the gradient of the log likelihood function:

$$\begin{aligned}\nabla \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) &= \sum_{i=1}^{n} (\mathbf{w}^T \mathbf{x}_i - \mathbf{y}_i)\mathbf{x}_i^T \\ &= \mathbf{w}^T \sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T - \sum_{i=1}^{n} \mathbf{y}_i \mathbf{x}_i^T\end{aligned}$$

The Normal Equations

Notice that $\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T$ is simply matrix multiplication, so if we write

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix},$$

The Normal Equations

then $\sum_{i=1}^{n} \mathbf{x}_i \mathbf{x}_i^T = \mathbf{X}^T\mathbf{X}$, $\sum_{i=1}^{n} \mathbf{y}_i \mathbf{x}_i^T = \mathbf{Y}^T\mathbf{X}$, and

The Normal Equations

$$\nabla \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) = \mathbf{w}^T\mathbf{X}^T\mathbf{X} - \mathbf{Y}^T\mathbf{X}.$$

The Normal Equations

The critical point is when $\nabla \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \sigma) = 0$. Solving for $\mathbf{w}$ yields

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}.$$

Fitting Polynomials

Not every dataset can be modeled using a simple line.

Many datasets follow a curved model which can be estimated using a **polynomial** function.

Fitting Polynomials

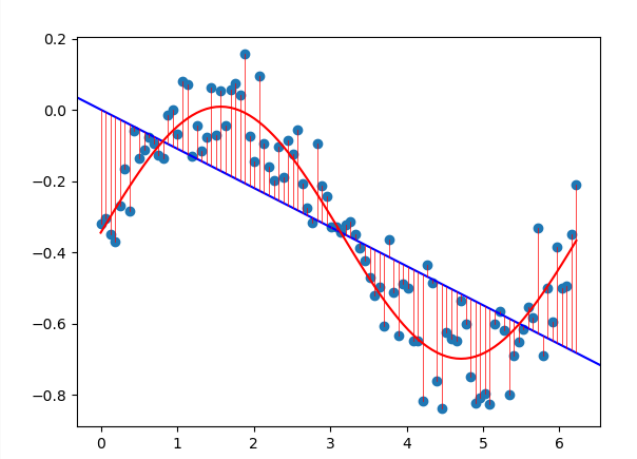

Figure 5: Data generated from a `sin` function (red).

Fitting Polynomials

Using a linear model will not fit the data generated from a nonlinear function.

What if we did not know that the data was generated using a sin function?

Fitting Polynomials

We can make our model more expressive by modifying the original input vectors using different degrees of polynomials.

Starting with x^3.

Fitting Polynomials

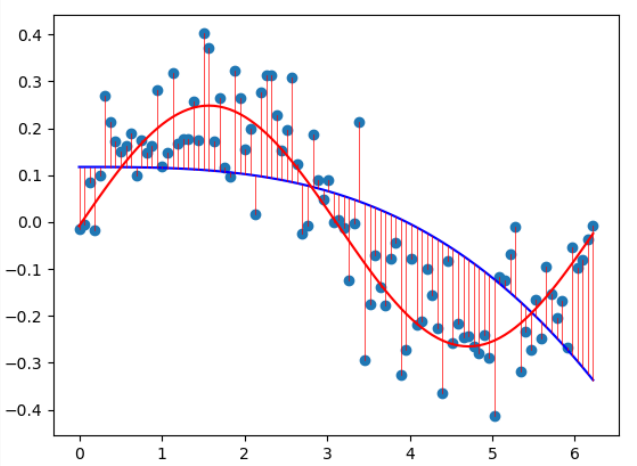

Figure 6: Fitting the data by raising the input to the power of 3.

Fitting Polynomials

This may have fit differently if the data was centered at $x = 0$.

Solution: add more degrees of freedom so that the model can fit more closely to the original.

Fitting Polynomials

The original input can be scaled up to any degree of polynomial.

$$\mathbf{X} = \begin{bmatrix} x_1 & x_1^2 & \cdots & x_1^m \\ x_2 & x_2^2 & \cdots & x_2^m \\ \vdots & \vdots & \ddots & \vdots \\ x_n & x_n^2 & \cdots & x_n^m \end{bmatrix}$$

Fitting Polynomials

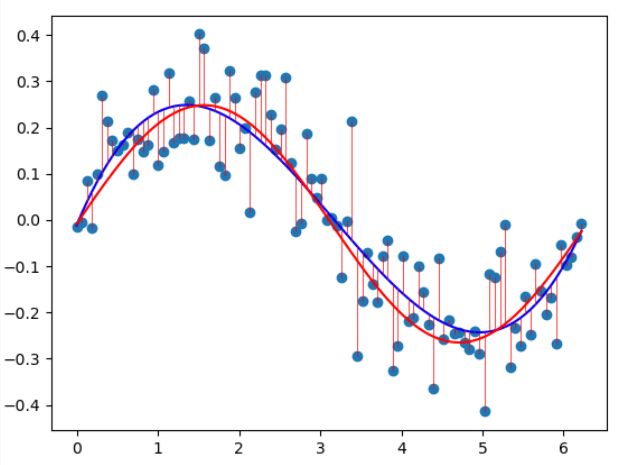

Figure 7: Fitting a third-degree polynomial to the data.

Linear Basis Function Models

We now consider creating a model that transforms the original input using one or more nonlinear functions.

This type of model is called a **linear basis function model**.

Linear Basis Function Models

Polynomial models are simply a specific implementation of linear basis function models.

Instead of using a linear combination of the inputs, we use a linear combination of basis functions.

Each basis function transforms the original input vectors $\mathbf{x}$.

Linear Basis Function Models

The model is then represented as

$$h(\mathbf{x}; \mathbf{w}) = \sum_{j=1}^{m} w_j \phi_j(\mathbf{x})$$

Linear Basis Function Models

Common basis functions are the sigmoid, Gaussian, or exponential function.

If we choose the sin function as a basis function, we can more closely fit our dataset using the least squares approach.

Fitting Polynomials

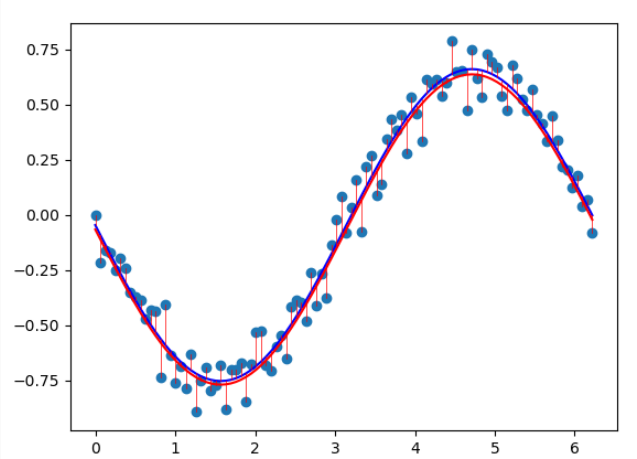

Figure 8: Fitting a sin function to the data.