NYU FRE 7773 - Week 2

Machine Learning in Financial Engineering

Ethan Rosenthal

Linear Models for Regression

Machine Learning in Financial Engineering

Ethan Rosenthal

Computer Setup?

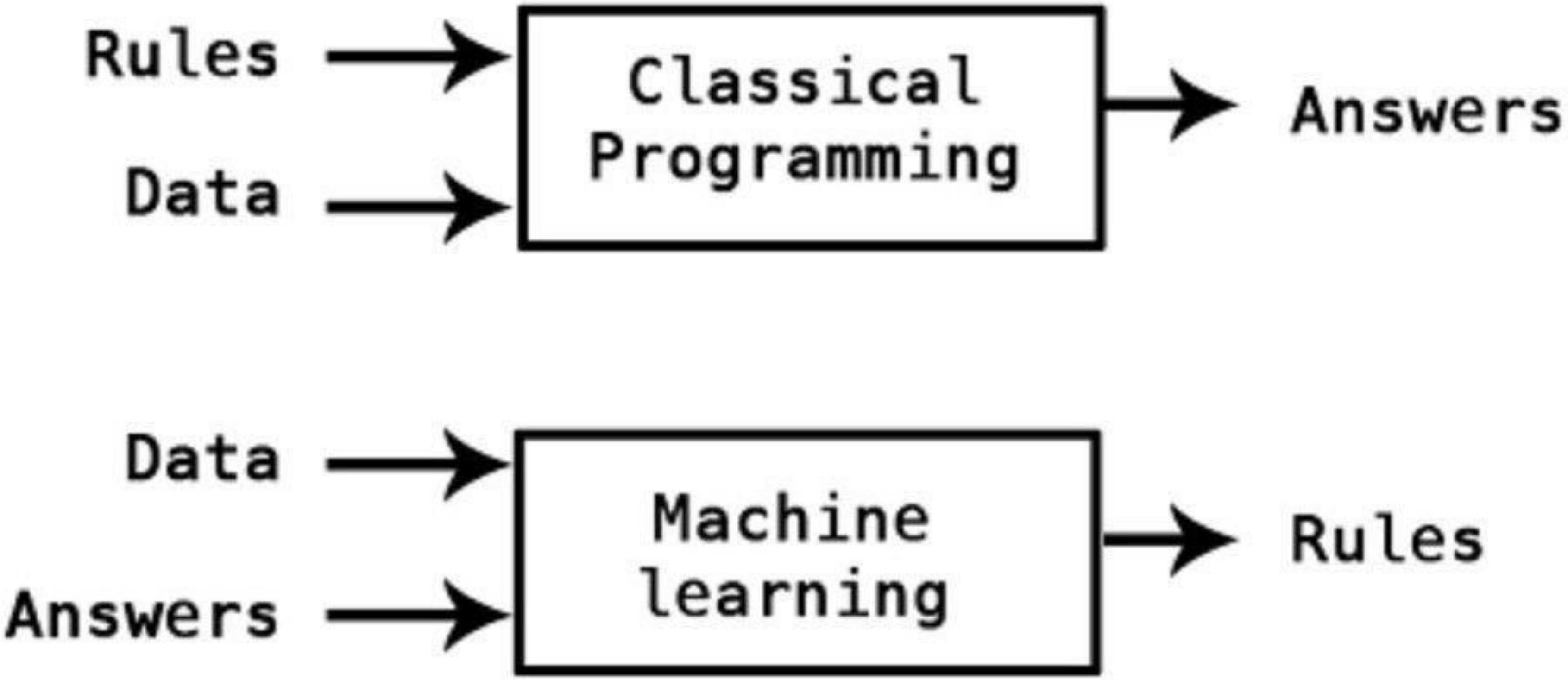
Linear Models for Regression

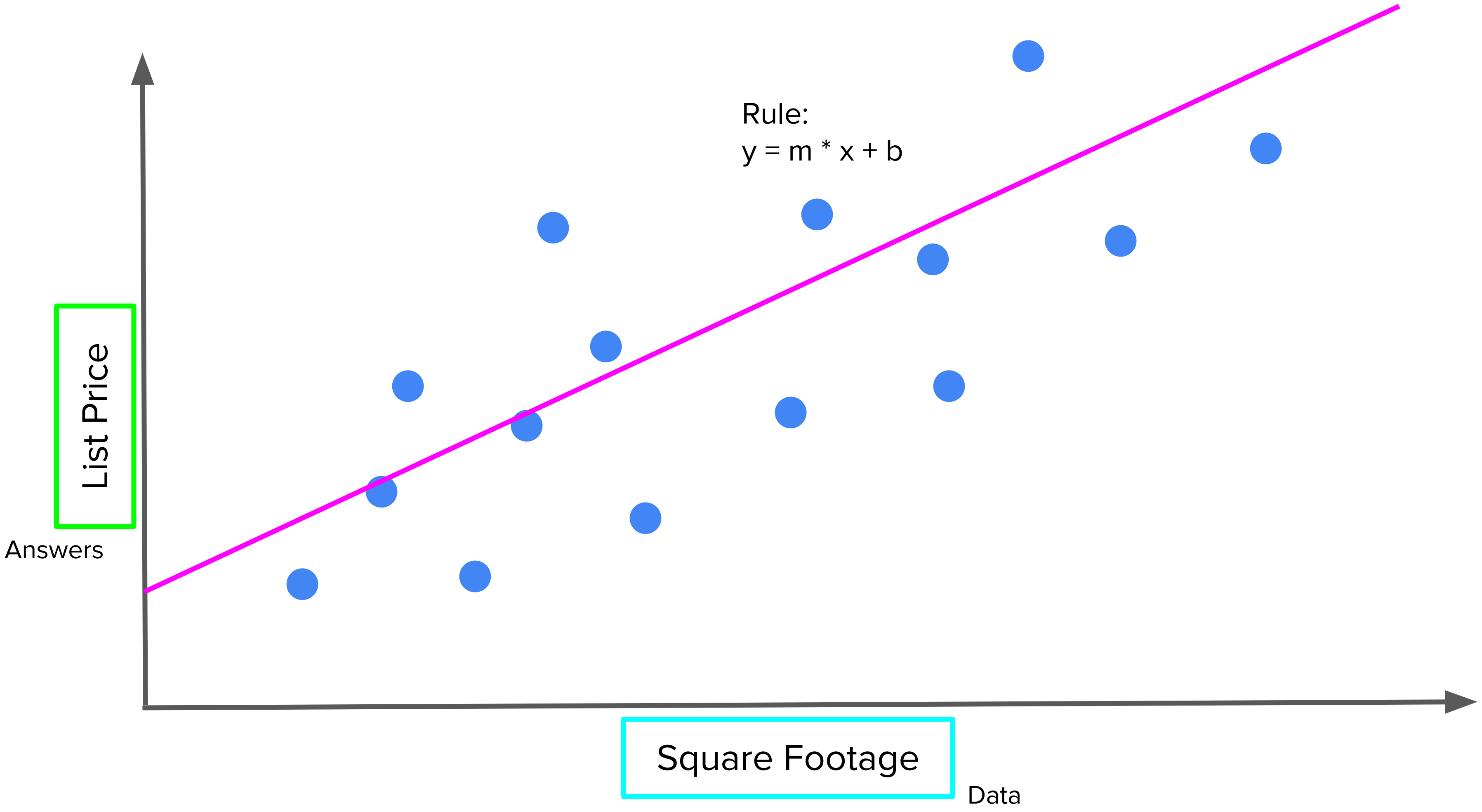
Machine Learning in Financial Engineering

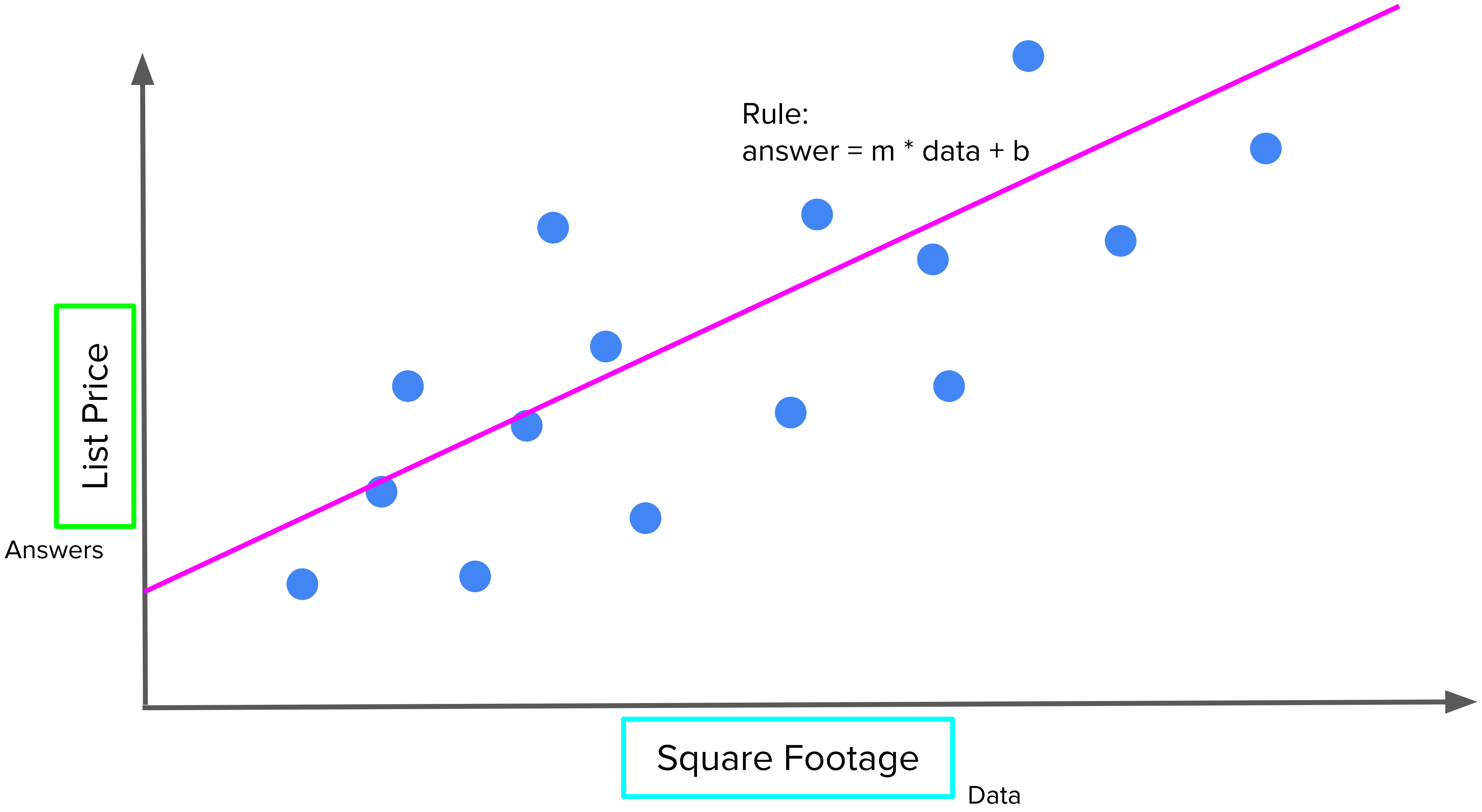
Ethan Rosenthal

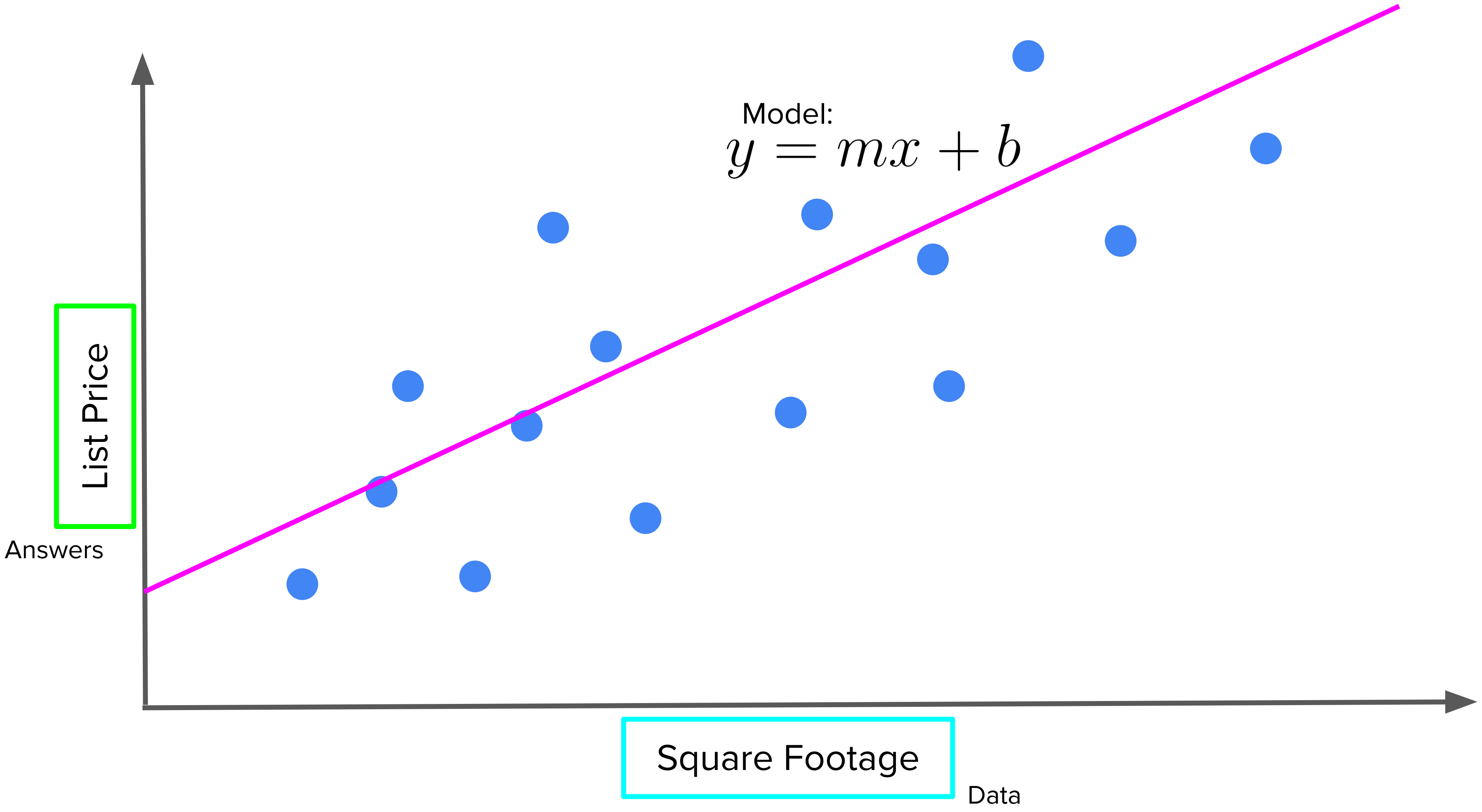
Linear Models for Regression

- **Regression**: for when we are predicting continuous values.
- **Linear Model**: a model that is linear in the model parameters/coefficients.


Rules
Data
Classical Programming
Answers
Data
Answers
Machine learning
Rules


Rule:
y = m * x + b
List Price
Answers
Square Footage
Data


Rule:
answer = m * data + b
List Price
Answers
Square Footage
Data


Model:
$y = mx + b$
List Price
Answers
Square Footage
Data

Let’s Generalize

- Imagine we have *n* samples or data points indexed by *i*.
- For each sample, we have *p* features (known “piece of information” about the sample) indexed by j.
- A linear model will look like:

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}$$

Where β_j denotes the *jth* model parameter/coefficient.

Let's Generalize

- Imagine we have *n* samples or data points indexed by *i*.
- For each sample, we have *p* features (known "piece of information" about the sample) indexed by j.
- A linear model will look like:

Known Data

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}$$

Where β_j denotes the *jth* model parameter/coefficient.

Known Answer

"Rules"
aka this is what we want to learn!

Let’s Generalize

Bias term (y-intercept)

- Imagine we have *n* samples or data points indexed by *i*.
- For each sample, we have *p* features (known “piece of information” about the sample) indexed by j.
- A linear model will look like:

Known Data

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}$$

Where β_j denotes the *jth* model parameter/coefficient.

Known Answer

“Rules”
aka this is what we want to learn!

Let's Generalize - single sample *i*

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}$$

Let's Generalize - single sample *i*

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + ... + \beta_p X_{ip}$$

$$y_i = \sum_{j=0}^{p} \beta_j X_{ij}$$

where we assume that $X_{i0} = 1$ (for mathematical convenience).

Let's Generalize - *n samples*

$$y_i = \sum_{j=0}^{p} \beta_j X_{ij}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix} = \begin{bmatrix} X_{10} & X_{11} & X_{12} & \dots & X_{1p} \\ X_{20} & X_{21} & X_{22} & \dots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots & \\ X_{n0} & X_{n1} & X_{n2} & \dots & X_{np} \end{bmatrix} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ . \\ . \\ . \\ \beta_p \end{bmatrix}$$

$$\vec{y} = \mathbf{X}\vec{\beta}$$

Let's Generalize - *n samples*

$$y_i = \sum_{j=0}^{p} \beta_j X_{ij}$$

$$\begin{bmatrix} y_1 \\ y_2 \\ . \\ . \\ . \\ y_n \end{bmatrix} = \begin{bmatrix} X_{10} & X_{11} & X_{12} & \dots & X_{1p} \\ X_{20} & X_{21} & X_{22} & \dots & X_{2p} \\ \vdots & \vdots & \ddots & \vdots & \\ X_{n0} & X_{n1} & X_{n2} & \dots & X_{np} \end{bmatrix} \cdot \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ . \\ . \\ . \\ \beta_p \end{bmatrix}$$

Known Data (features)

$$\vec{y} = \mathbf{X}\vec{\beta}$$

"Rules"
aka this is what we want to learn!

Known Answers

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.
4. Determine the model parameters that produce the minimum loss.

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.
4. Determine the model parameters that produce the minimum loss.

Housekeeping:

The model *predictions* are often denoted $\hat{y}_i$

The *ground truth / answers* are often denoted y_i (no hat).

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.

$$\hat{\vec{y}} = \mathbf{X}\vec{\beta}$$

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.

Mean Squared Error (MSE):

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \vec{X}_i \cdot \vec{\beta}\right)^2$$

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{j=0}^{p} X_{ij}\beta_j\right)^2$$

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.

Mean Squared Error (MSE):

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

true value

prediction

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \vec{X}_i \cdot \vec{\beta}\right)^2$$

data

Unknown!

$$\mathcal{L} = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \sum_{j=0}^{p} X_{ij}\beta_j\right)^2$$

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.
4. Determine the model parameters that produce the minimum loss.
    a. Take the derivative of your loss function with respect to the model parameters.
    b. Set it equal to zero.
    c. Solve for the model parameters.

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.
4. Determine the model parameters that produce the minimum loss.
    a. Take the derivative of your loss function with respect to the model parameters.
    b. Set it equal to zero.
    c. Solve for the model parameters.

$$\frac{\partial \mathcal{L}}{\partial \vec{\beta}} = 0$$

The ML Recipe

1. Think up some model
2. Feed **data** into the model and make predictions.
3. Calculate the loss between predictions and true values.
4. Determine the model parameters that produce the minimum loss.
   a. Take the derivative of your loss function with respect to the model parameters.
   b. Set it equal to zero.
   c. Solve for the model parameters.

$$\frac{\partial \mathcal{L}}{\partial \vec{\beta}} = 0$$

$$\vec{\beta} = \left(\mathbf{X}^T \mathbf{X}\right)^{-1} \mathbf{X}^T \vec{y}$$

The ML Recipe - Meta Comments

1. Think up some model
    a. We chose a linear model. There are many others!
2. Feed **data** into the model and make predictions.
    a. We chose what features to use.
3. Calculate the loss between predictions and true values.
    a. We chose Mean Square Error as our loss. There are many others!
4. Determine the model parameters that produce the minimum loss.
    a. We analytically determined the parameters. There are many other ways to determine them!
    b. Also, sometimes you cannot analytically determine the parameters.

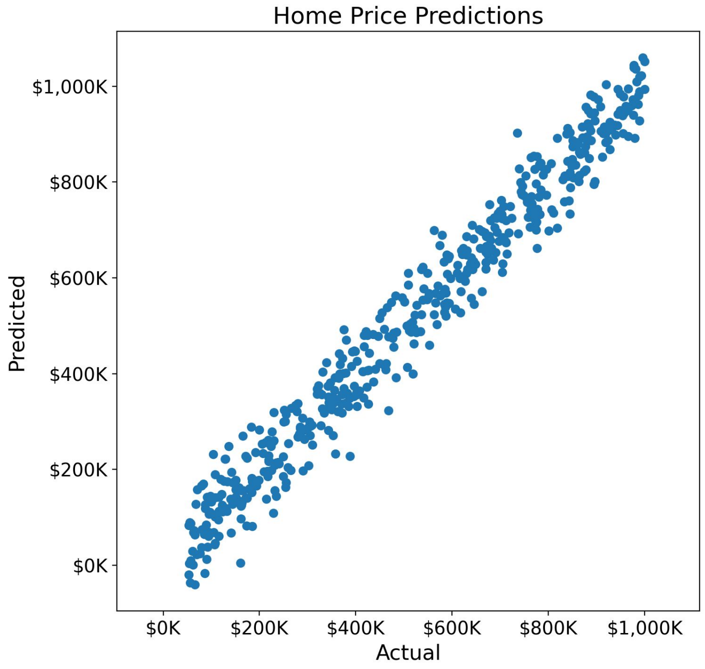
The ML Recipe - Meta Comments

1. Think up some model
    a. We chose a linear model. There are many others!
2. Feed **data** into the model and make predictions.
    a. We chose what features to use.
3. Calculate the loss between predictions and true values.
    a. We chose Mean Square Error as our loss. There are many others!
4. Determine the model parameters that produce the minimum loss.
    a. We analytically determined the parameters. There are many other ways to determine them!
    b. Also, sometimes you cannot analytically determine the parameters.

While we denote step 1 as “the model”, all of these steps involve choices that impact the eventual model that is used for prediction.

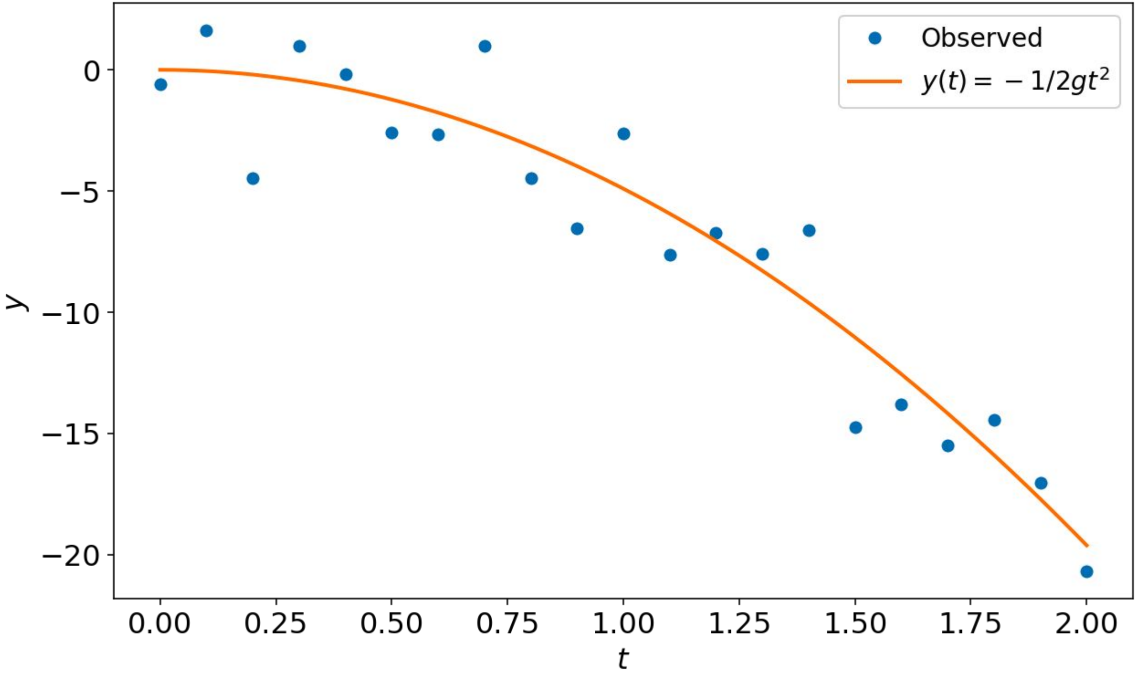
Inference vs. Prediction

- So far, we have focused on being able to build a model that predicts things.



Inference vs. Prediction

- Inference: inspect your model to learn about your system.



Inference vs. Prediction

- Inference: inspect your model to learn about your system.
- Examples:
    - Which feature is most important for accurate predictions?
    - If I increase feature Z by 10%, how will that change the prediction?
    - What is the uncertainty in my prediction?
    - What is the uncertainty in a model parameter?

Inference vs. Prediction

- Inference: inspect your model to learn about your system.
- Examples:
    - Which feature is most important for accurate predictions?
    - If I increase feature Z by 10%, how will that change the prediction?
    - What is the uncertainty in my prediction?
    - What is the uncertainty in a model parameter?
- This is often more difficult and requires statistical guarantees.
- This is well-studied for linear models but not so for more complicated models.
- There is an inherent tradeoff between model complexity and interpretability.
- Model complexity can be correlated with model *performance*, so there can be a tradeoff between performance and interpretability.

Nonlinear Features for Linear Models

- We can do whatever we want to the features **X**.

$$y_i = \sum_{j=0}^{p} \beta_j X_{ij}$$

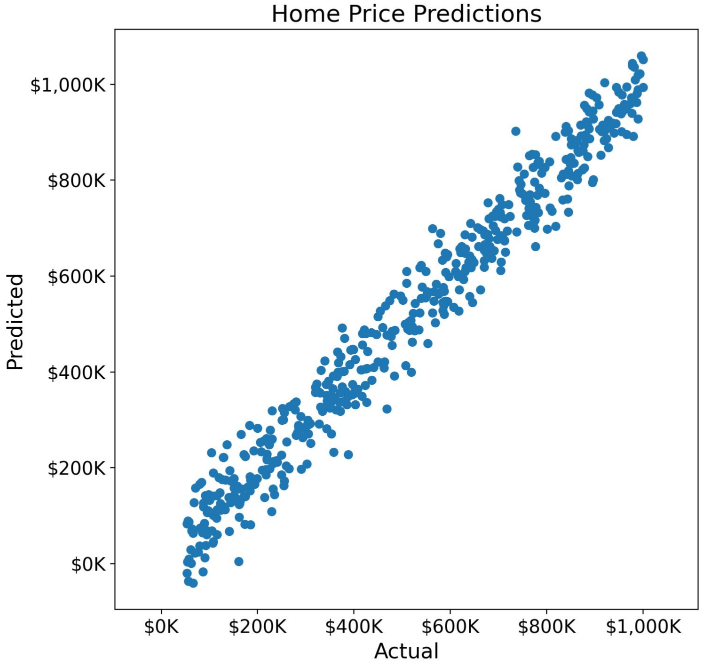
- We can square a feature, we can multiple features by each other, we can apply a sine, etc...

$$y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i1}^2 + \beta_3 X_{i1} X_{i2} + \beta_4 sin\left(X_{i3}\right)$$

- While these features are nonlinear, the model is linear in the parameters β.

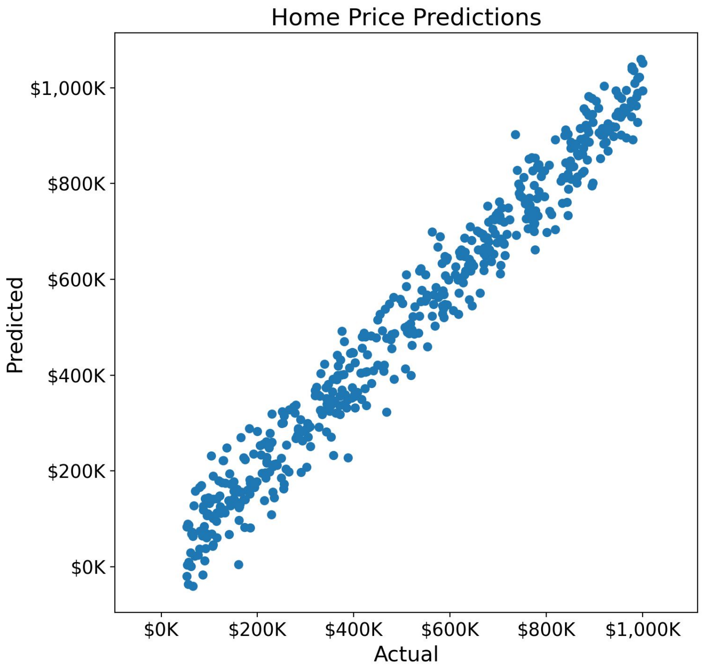
Evaluating Regression Models

How do we know how good our prediction is?



How do we know how good our prediction is?

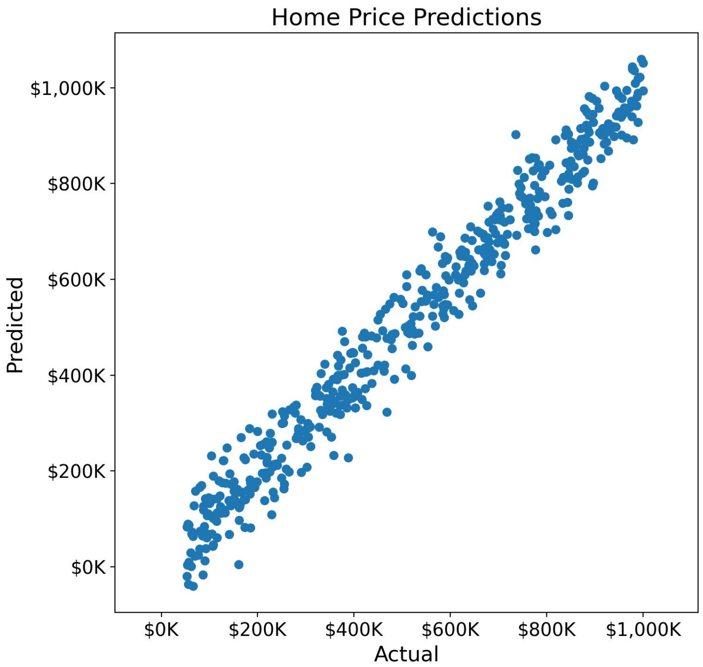
- The model's performance is based on the context in which the model is used.
- We often try to collapse "performance" down to a single number.
- There are many statistical measures of model quality which may or may not be useful in practice.



How do we know how good our prediction is?

- If we are minimizing the Mean Squared Error (MSE), then we can use MSE as a performance metric.
- Outliers will have a large effect on this metric.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$



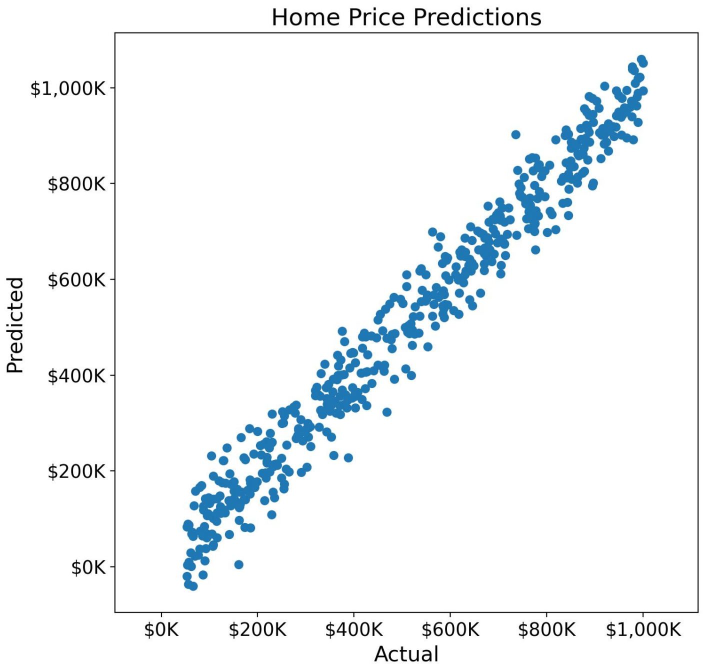
Regression Metrics

- Mean Squared Error - minimizes outliers, good statistical properties, hard to interpret.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

- Mean Absolute Error - easy to interpret, all loss is treated equally, not robust to outliers.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$



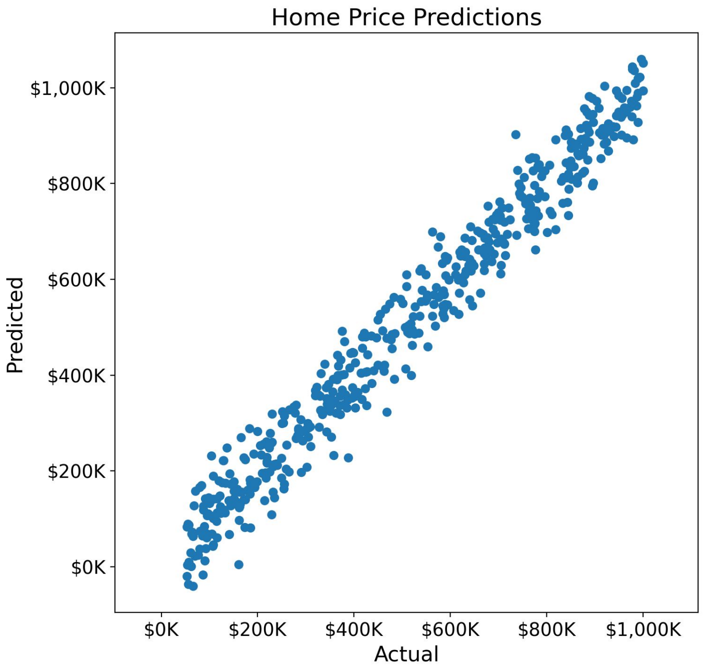
Regression Metrics

- $\mathcal{R}^2$/ coefficient of determination - statistically interpretable as the proportion of variance that's predictable

$$\mathcal{R}^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

- Mean Absolute Percentage Error - interpretable, error is independent of scale.

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$



Scientific Computing