Regression

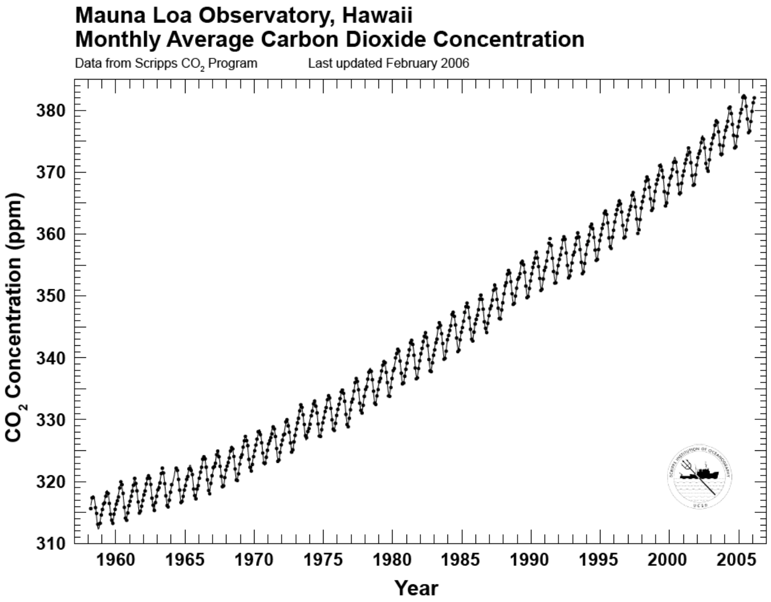
Machine Learning and Computational Statistics (DSC6135)

Objectives

- Understand steps of a **regression** task (training, prediction, evaluation)
- Non-parametric/parametric linear regression
- Probabilistic perspective

1. Introduction to Regression

Supervised learning task



Goal: Predict *continuous* output y given inputs x

- input x_i: also called features, covariates, predictors, attributes
- output y_i: also called responses or labels

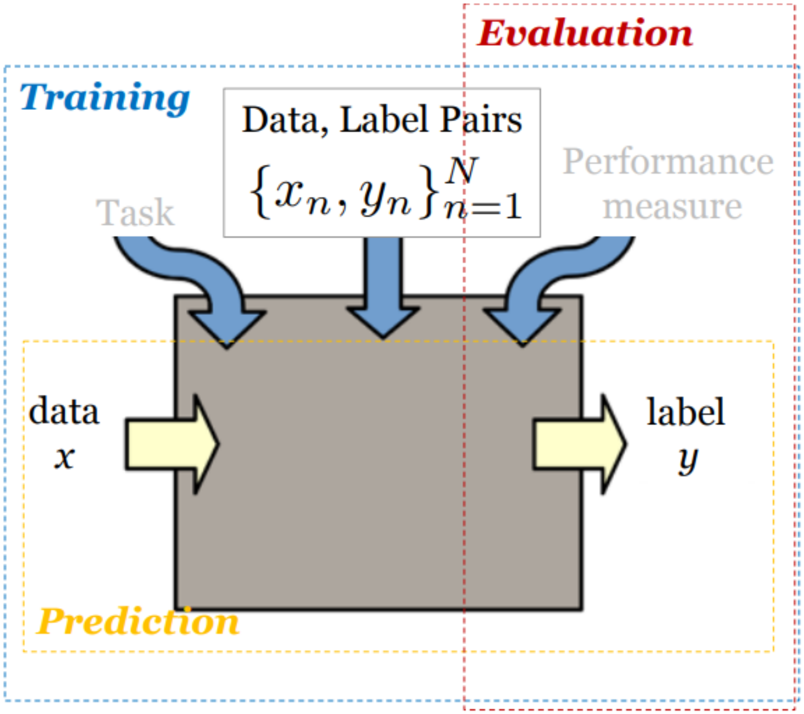
Other examples:

1.Predicting a person’s height given the height of their parents.

2.Predicting the amount of time someone will take to pay back a loan given their credit history.

3.Predicting what time a package will arrive given current weather and traffic conditions.

1. Introduction to Regression

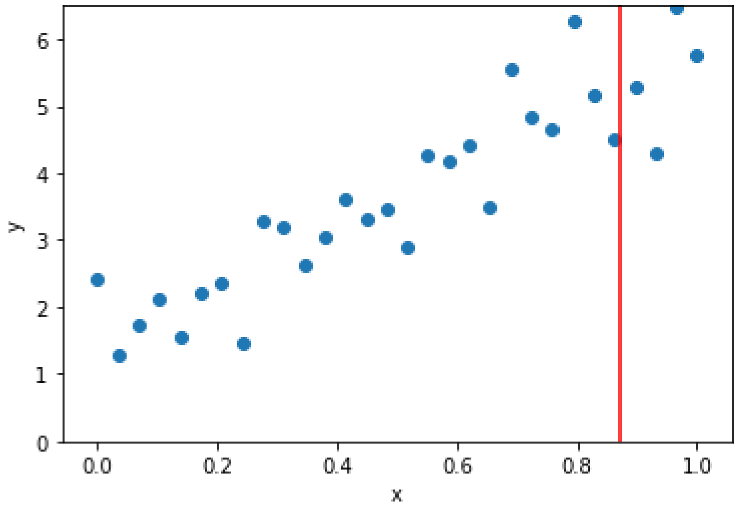


Supervised learning task:

- *Training step*: given $\{x_n, y_n\}_{n=1}^N$, learn a predictive **function**
- *Evaluation Step*: given $\{\hat{y}_n, y_n\}_{n=1}^N$, measure error/quality
- *Prediction step*: given features $x^\star$, predict response $y^\star$

1. Introduction to Regression

In [14]:
```
x,y = plot_lr_example()
```



Which value would you predict at $x^{\star}$ (the red line)?

Which prediction is better?

In the following...

1. how to train and get new predictions
2. how to evaluate

...for different methods.

Non-parametric regression: A non-parametric model simply means that we don't make any assumptions about the form of our data. We only need to use the data itself to make predictions.

2. Non-parametric Regression: k-nearest neighbors

A simple prediction step: look at your neighbors!

1. Find k nearest points $\{x_1, \ldots, x_k\}$ to $x^\star$
2. Predict $\hat{y} = \frac{1}{k} \sum_{j=1}^{k} y_j$

What about the *training step*?

Exercise: program k-NN function

(HW0 at: [https://melaniefp.github.io/intro_to_ML_DSC6135/ (https://melaniefp.github.io/intro_to_ML_DSC6135/)](https://melaniefp.github.io/intro_to_ML_DSC6135/))

In [10]:

```python
def predict_knn_regression(x, x_observed, y_observed, k=1):
    '''
    Function to predict output y for input x given past data
    (x_observed, y_observed) in 1-dimension
    Parameters:
        x_observed: (N_obs,) numpy array, inputs observed in the past
        y_observed: (N_obs,) numpy array, outputs observed in the past
        x: (N,) numpy array, inputs of interest
        k: scalar, number of neighbors to consider
    Return:
        y: (N,) numpy array, outputs of interest
    '''
    # COMPLETE

    return y
```

A simple idea: k-nearest neighbors

What is good? What is bad of this approach?

Advantages:

```
Very flexible! No need to assume any function class (e.g., line)
```

Inconvenients:

```
(a) to make prediction, needs to keep all data (memory issues)
(b) needs to choose a distance
(c) high-dimensional issues (curse of dimensionality)
```

3. Parametric Regression

Assume a function class f and its parameters w :

$$y = f_w(x)$$

General formula for *training step*:

1. Choose a class $f_w(.)$ (e.g., linear, quadratic, polynomial, etc...)
2. Choose a loss or objective function $\mathcal{L}(w)$
3. Pick best $w^\star = \min_w \mathcal{L}(w)$

In particular: Linear Regression

For example, in the case of linear regression,

1. Choose a class $f_{\mathbf{w}}(.)$

$$y = f_{\mathbf{w}}(\mathbf{x}) = w_0 + w_1 x_1 + \ldots + w_D x_D = \mathbf{w}^T \mathbf{x}$$

(to simplify notation, we assume that $x_0 = 1$)

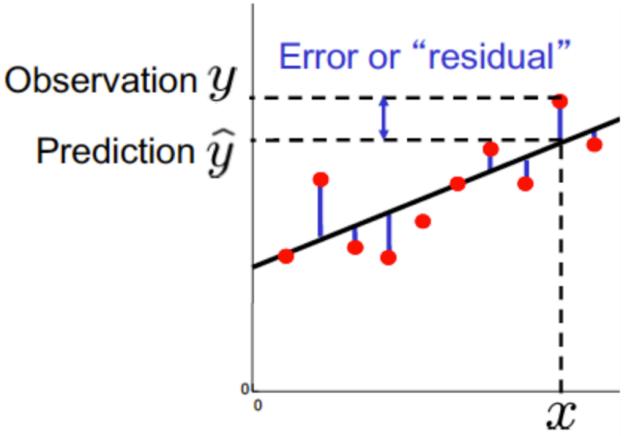
1. Choose a loss $\mathcal{L}(w)$

- mean squared error

$$\mathrm{MSE} = \frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)^2$$

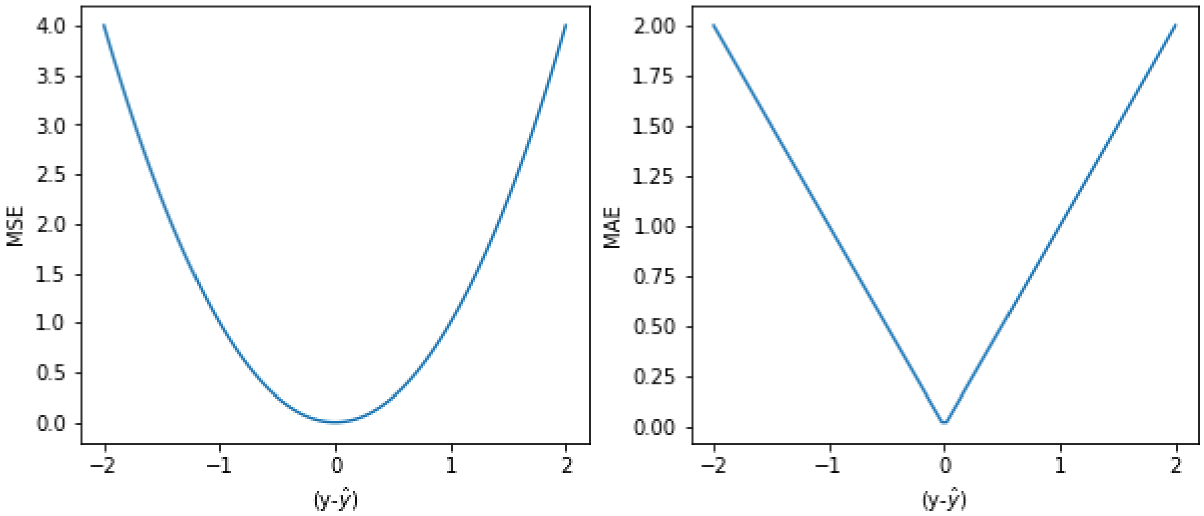
- mean absolute error

$$\mathrm{MAE} = \frac{1}{N}\sum_{n=1}^{N}|y_n - \hat{y}_n|$$



In [11]:
```
fig, ax = plt.subplots(1,2,figsize=(10,4))
x = np.linspace(-2,2,100)
y_MSE = x**2
y_MAE = np.abs(x)
ax[0].plot(x,y_MSE); ax[0].set_xlabel('(y-$\hat{y})$')
ax[0].set_ylabel('MSE')
ax[1].plot(x,y_MAE); ax[1].set_xlabel('(y-$\hat{y})$')
ax[1].set_ylabel('MAE')
```

Out[11]:
```
Text(0, 0.5, 'MAE')
```



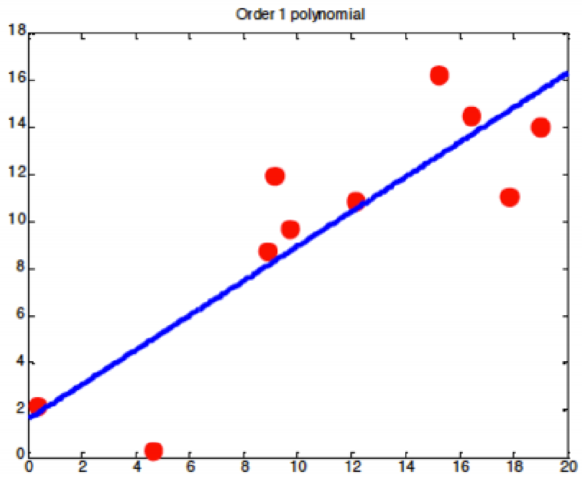
- Which error metric is more sensitive to outliers?
- When would you choose one metric versus another?

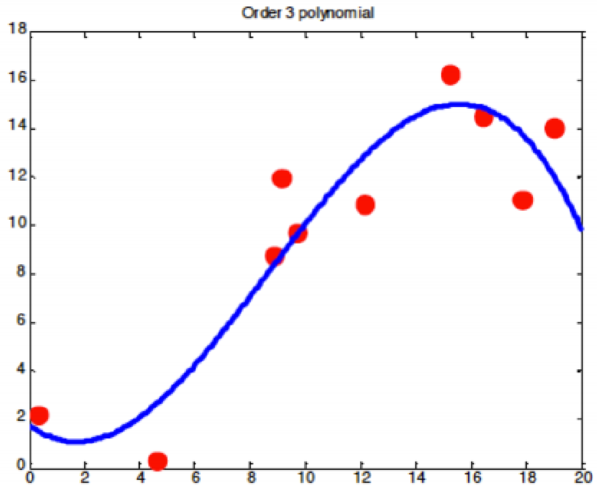
1. Pick best
   $$w^\star = \min_w \mathcal{L}(w)$$

This is usually hard (requires optimization) -- in the case of linear regression, analytic solution exists!

Analytic approach (see blackboard)

Polynomial regression





Are we limited by lines? No! We can use basis regression:

$$y = \mathbf{w}^T \phi(\mathbf{x})$$

For example...

$$\phi(x) = [x,\ \ x^2,\ \ x^3]$$

What feature transform to use?

- sin / cos for periodic data
- polynomials for high-order dependencies

$$\phi(x_i) = [x_i,\ x_i^2,\ x_i^3]$$

- interactions between feature dimensions

$$\phi(x_i) = [x_{i1}x_{i2},\ x_{i3}x_{i4}]$$

- Many other choices possible

In general, all these models are called **kernelized** linear regression, or **basis** linear regression.

EXERCISE:

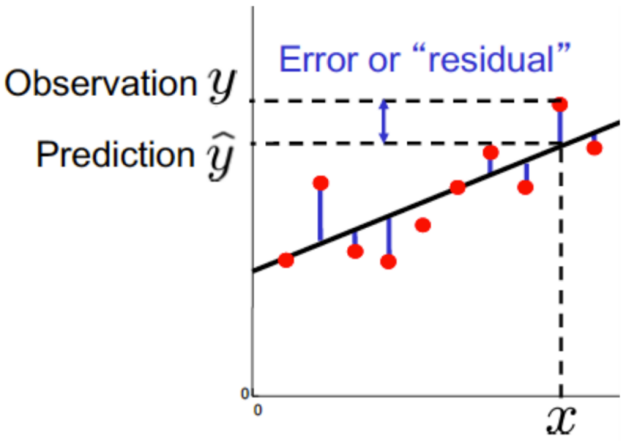
Question: Plot (x,y) in the range [-5,5] where $y = \mathbf{w}^T \phi(\mathbf{x}), \mathbf{w} = [1, -0.4, 0.2]$ and $\phi(\cdot)$ are polynomial basis.

(tip: use lambda functions)

Noisy function!

Question: Plot (x,y) in the range [-5,5] where $y = \mathbf{w}^T \phi(\mathbf{x}) + \epsilon, \mathbf{w} = [1, -0.4, 0.2]$ and $\phi(\cdot)$ are polynomial basis. (assuming Gaussian noise)

Question: what does it mean to pick a loss function?



5. Probabilistic perspective of Regression

Probabilistic Interpretation: Modeling of noise. We make a `story` for how the data was created, this is called a `generative model`, this is really useful to understand your assumptions!

Example:

$$y = \mathbf{w}^T \mathbf{x} + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \sigma^2)$$

Likelihood of the model: $p(data|model\ \ parameters)$

Let's write down the likelihood (blackboard)

Maximum Likelihood Estimation for Bayesian Linear Regression

Let us define $\beta = \frac{1}{\sigma^2}$ as the inverse of the variance, or precision. The likelihood of our data set is given by:

$$p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(\mathbf{w}^T\mathbf{x}_n, \beta^{-1})$$

We then take the logarithm of the likelihood, and since the logarithm is a strictly increasing, continuous function, this will not change our optimal weights $\mathbf{w}$:

$$\ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(\mathbf{w}^T\mathbf{x}_n, \beta^{-1})$$

Using the density function of a univariate Gaussian:

$$\begin{aligned}
\ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \beta) &= \sum_{n=1}^{N} \ln \frac{1}{\sqrt{2\pi\beta^{-1}}} e^{-(y_n - \mathbf{w}^T\mathbf{x}_n)^2/2\beta^{-1}} \\
&= \frac{N}{2} \ln \beta - \frac{N}{2} \ln(2\pi) - \frac{\beta}{2} \sum_{n=1}^{N} (y_n - \mathbf{w}^T\mathbf{x}_n)^2
\end{aligned}$$

Notice that this is a quadratic function in $\mathbf{w}$, which means that we can solve for it by taking the derivative with respect to \textbf{w}, setting that expression to 0, and solving for $\mathbf{w}$:

$$\frac{\partial \ln p(\mathbf{Y}|\mathbf{X}, \mathbf{w}, \beta)}{\partial \mathbf{w}} = \beta \sum_{n=1}^{N} (y_n - \mathbf{w}^T \mathbf{x}_n)\mathbf{x}_n^T$$

$$0 = \beta \sum_{n=1}^{N} (y_n - \mathbf{w}^T \mathbf{x}_n)\mathbf{x}_n^T$$

$$0 = \sum_{n=1}^{N} y_n \mathbf{x}_n^T - \mathbf{w}^T \sum_{n=1}^{N} \mathbf{x}_n \mathbf{x}_n^T$$

Notice that this is exactly the same form as the analytical expression of linear regression with mean square error (MSE) loss function. Solving for $\mathbf{w}$ as before:

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

4. Quick review of probability distributions

Discrete Random variable: function defined on set of all possible outcomes thaat assigns a probability value for each outcome

$$X : \Omega \to \mathbb{R}$$

Examples:

- Coin flip: head or tails?
- Dice roll: 1 or 2 or ... 6?

Probability mass function:

- X is the random variable
- x is a particular observed value
- $p(X = x)$ is the probability of observation

Expected value

$$\mathbb{E}[X] = \sum_{x} p(X = x)x$$

Exercise:

```
* draw pmf for a normal 6-sided dice roll
* draw pmf of a dice with 2 sides with value 1, and 0 sides with value 2
```

Exercise:

		X	
		Candidate A	Candidate B
Y	Young voters	0.28	0.42
	Senior voters	0.24	0.06

Joint probability

$$p(X = \text{"candidate A"}, Y = \text{"young voters"}) = ?$$

Marginal probability

$$p(X = \text{"candidate B"}) = ?$$

Conditional probability

$$p(Y = \text{"senior voters"} | X = \text{"candidate A"}) = ?$$

Rules of Probability

sum rule

$$p(X) = \sum_{Y} p(X, Y)$$

product rule

$$p(X, Y) = p(Y|X)p(X) = p(X|Y)p(Y)$$

Continuous Random Variables

Any random variable whose possible outcomes are not a discrete set, but take values on a number line. Examples include:

- uniform draw between 0 and 1
- Gaussian draw

Continuous Random Variables

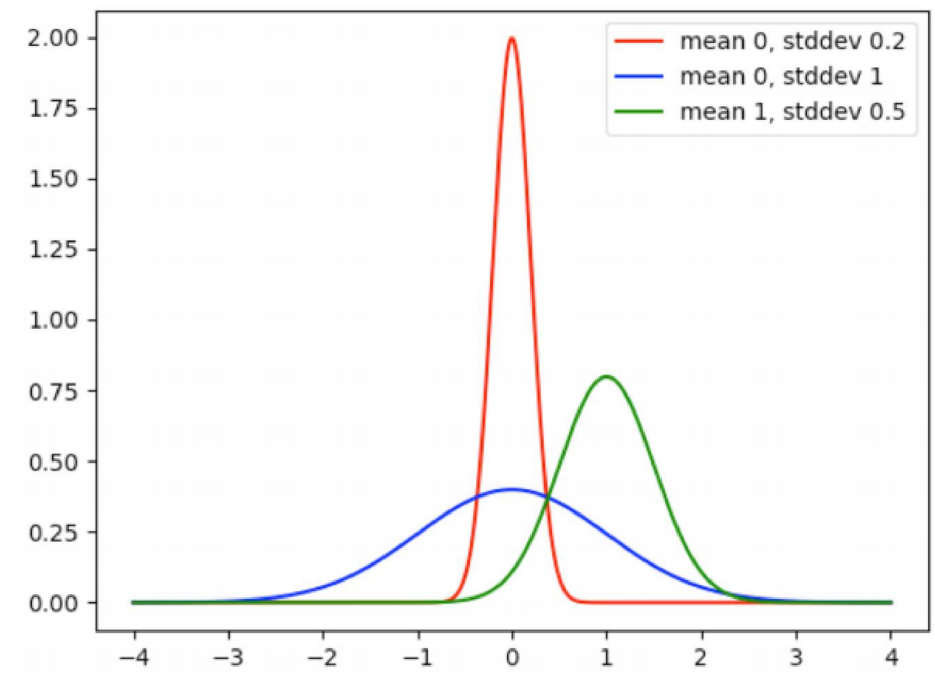
Probability Density Function

- Generalizes probability mass function for discrete random variable to continuous. Notation: by convention, we denote "small" x to the random variable.
- Any pdf $p(x)$ must satisfy two properties:

$$\forall x : p(x) \geq 0$$

$$\int_x p(x)dx = 1$$

Example: plots for Gaussian pdf



What do you notice about y-axis values...

Is there a problem here?

Value $p(x)$ can take any positive value, you should NOT interpret as "probability of drawing exactly x", instead, you should interpret as "density at vanishing small interval around x".

Discussion

We have seen kNN regression, linear regression, polynomial regression.

- What are the pros and cons of each?
- How to interpret?

Summary

- **kNN regression**:
  - function class: piece-wise constant
  - design choices: number of neighbors, distance metric, how neighbors vote
  - how to interpret? inspect neighbors
- **linear regression**:
  - function class: linear
  - design choices: bias or not
  - how to interpret? inspect weights
- **polynomial or basis regression**:
  - function class: flexible, depends on basis
  - design choices: bias or not; basis parameters
  - how to interpret? inspect weights

Glossary

Regression: A class of techniques that seeks to make predictions about un- known continuous target variables given observed input variables.

Linear Regression: Suppose we have an input $\mathbf{x} \in \mathbb{R}^D$ and a continuous target $y \in \mathbb{R}$. Linear regression determines weights $w_i \in \mathbb{R}$ that combine the values of x_i to produce y:

$$y = w_0 + w_1 x_1 + \ldots + w_D x_D$$

Objective Function: A function that measures the 'goodness' of a model. We can optimize this function to identify the best possible model for our data.

Residual: The residual is the difference between the target y and predicted $\hat{y} = f_w(x)$ value that a model produces

Basis Function: Typically denoted by the symbol φ(·), a basis function is a transformation applied to an input data point x to move our data into a different input domain.