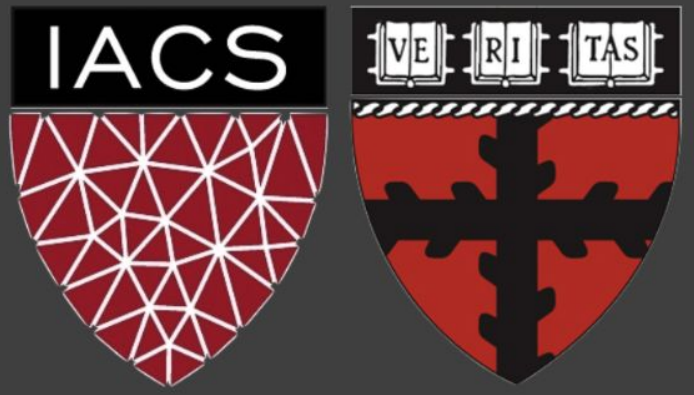
Introduction to Regression

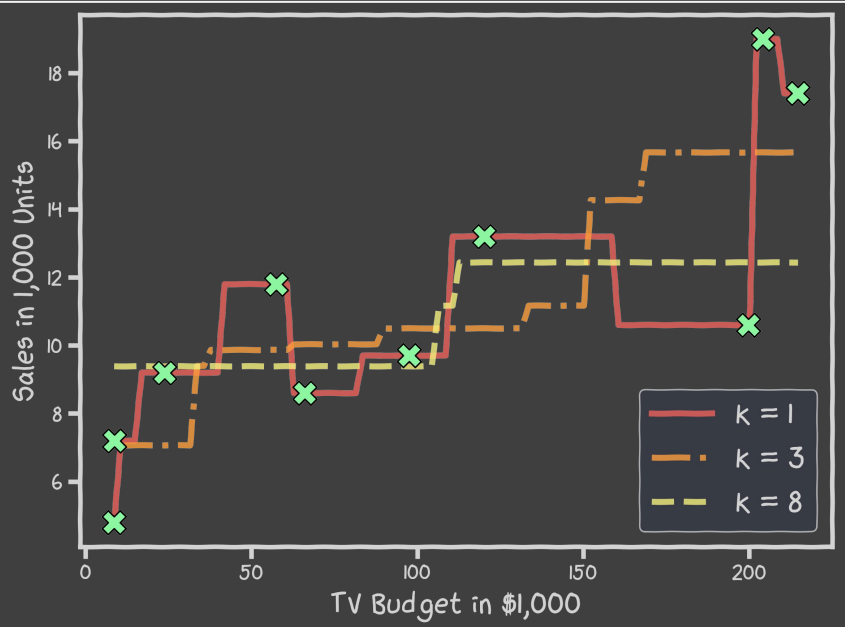
Part A - kNN

IACS-MACI Internship

Pavlos Protopapas



Linear Models



Note that in building our kNN model for prediction (non-parametric), we did not compute a closed form for $\hat{f}$.

What if we ask the question:

"how much more sales do we expect if we double the TV advertising budget?"

Linear Models

We can build a model by first assuming a simple form of f:

$$f(x) = \beta_0 + \beta_1 X$$

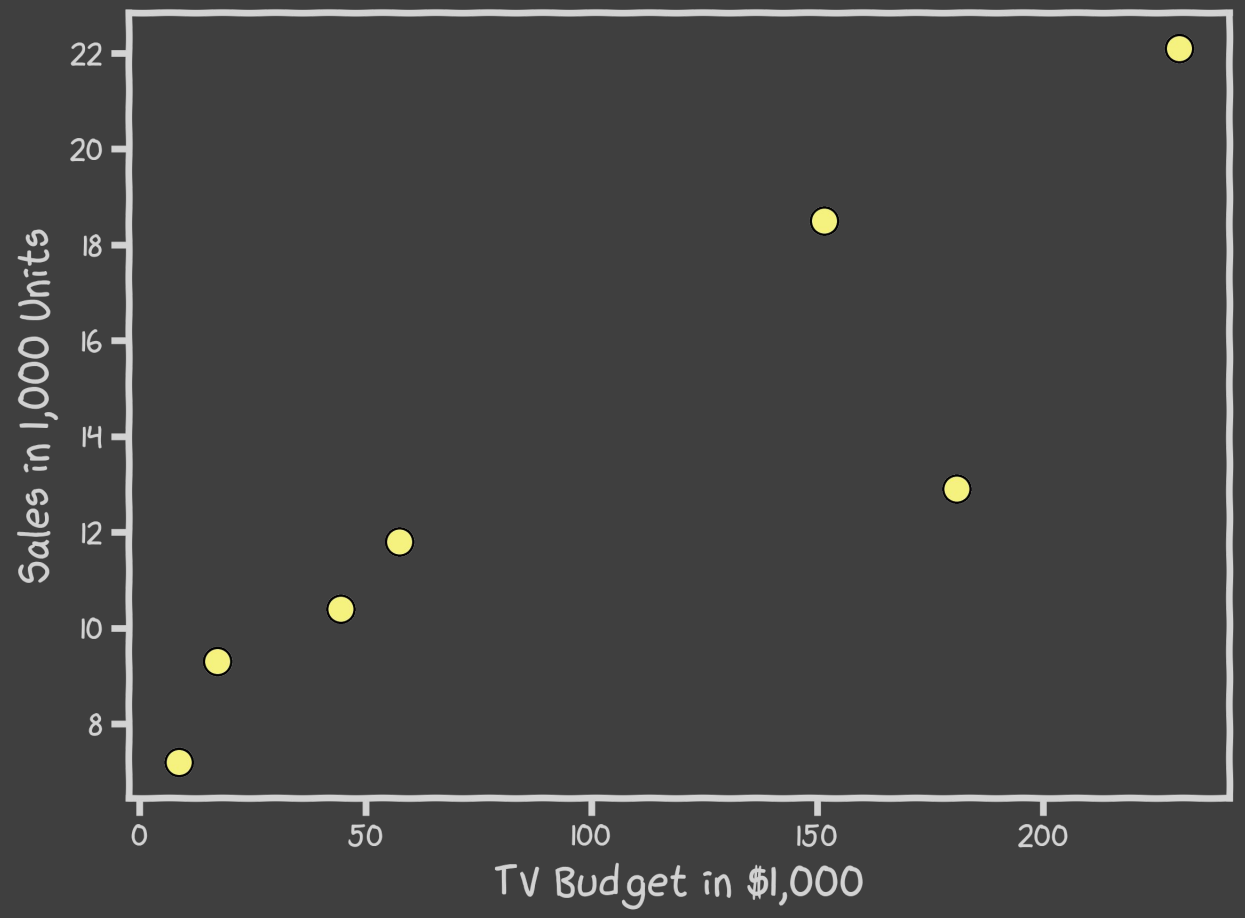
… then it follows that our estimate is:

$$\hat{Y} = \hat{f}(x) = \hat{\beta}_0 + \hat{\beta}_1 X$$

where $\hat{\beta}_1$ and $\hat{\beta}_0$ are **estimates** of β_1 and β_0 respectively, that we compute using observations.

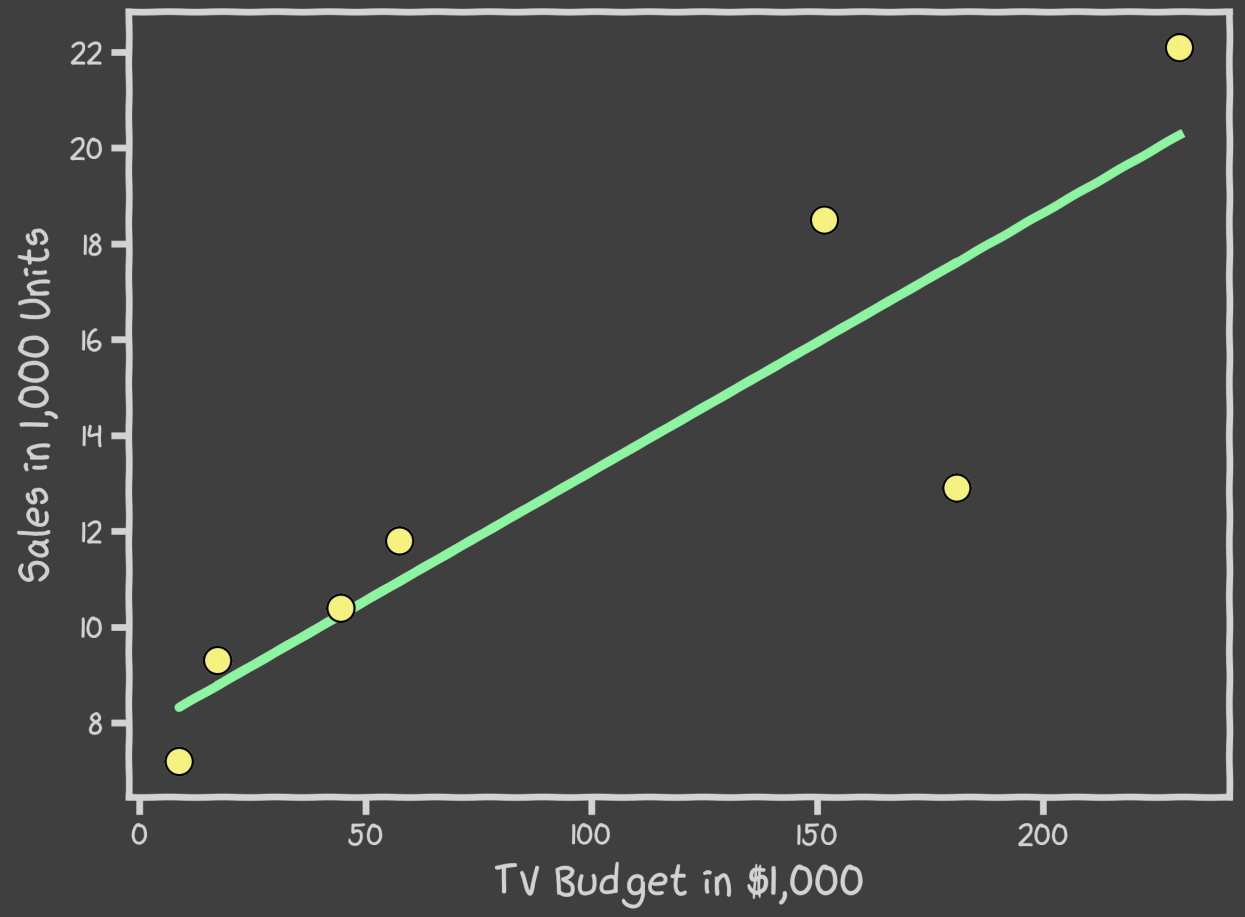
Estimate of the regression coefficients

For a given data set



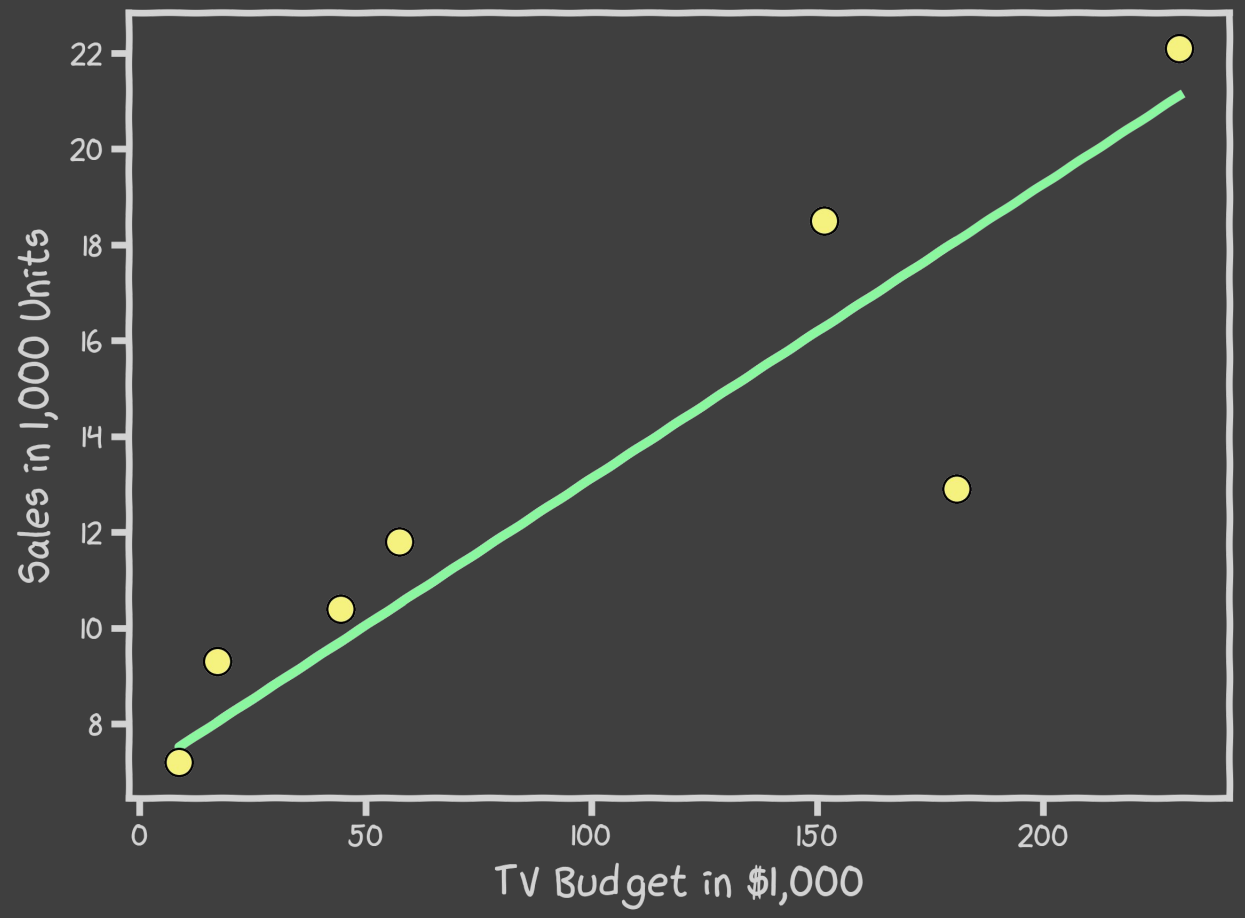
Estimate of the regression coefficients (cont)

Is this line good?



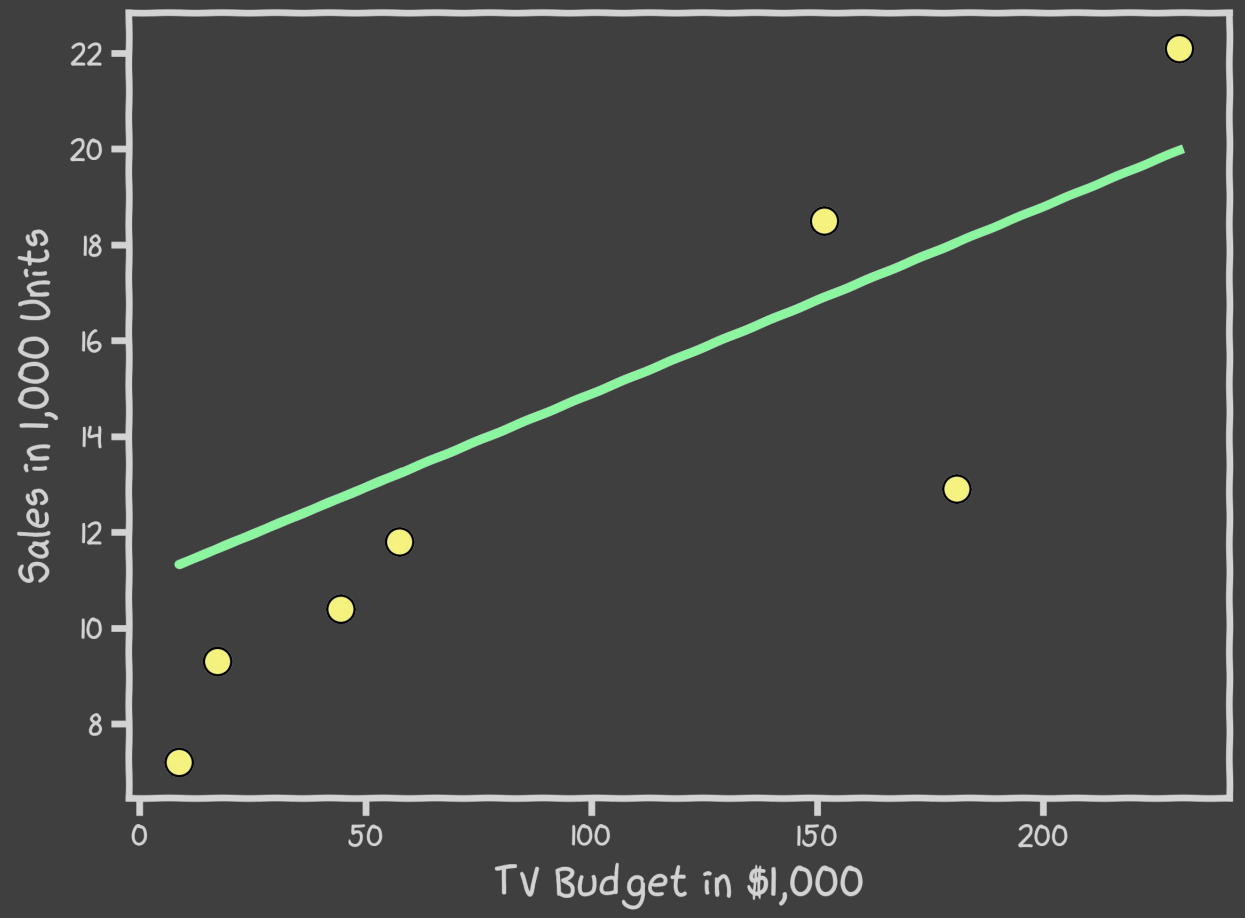
Estimate of the regression coefficients (cont)

Maybe this one?



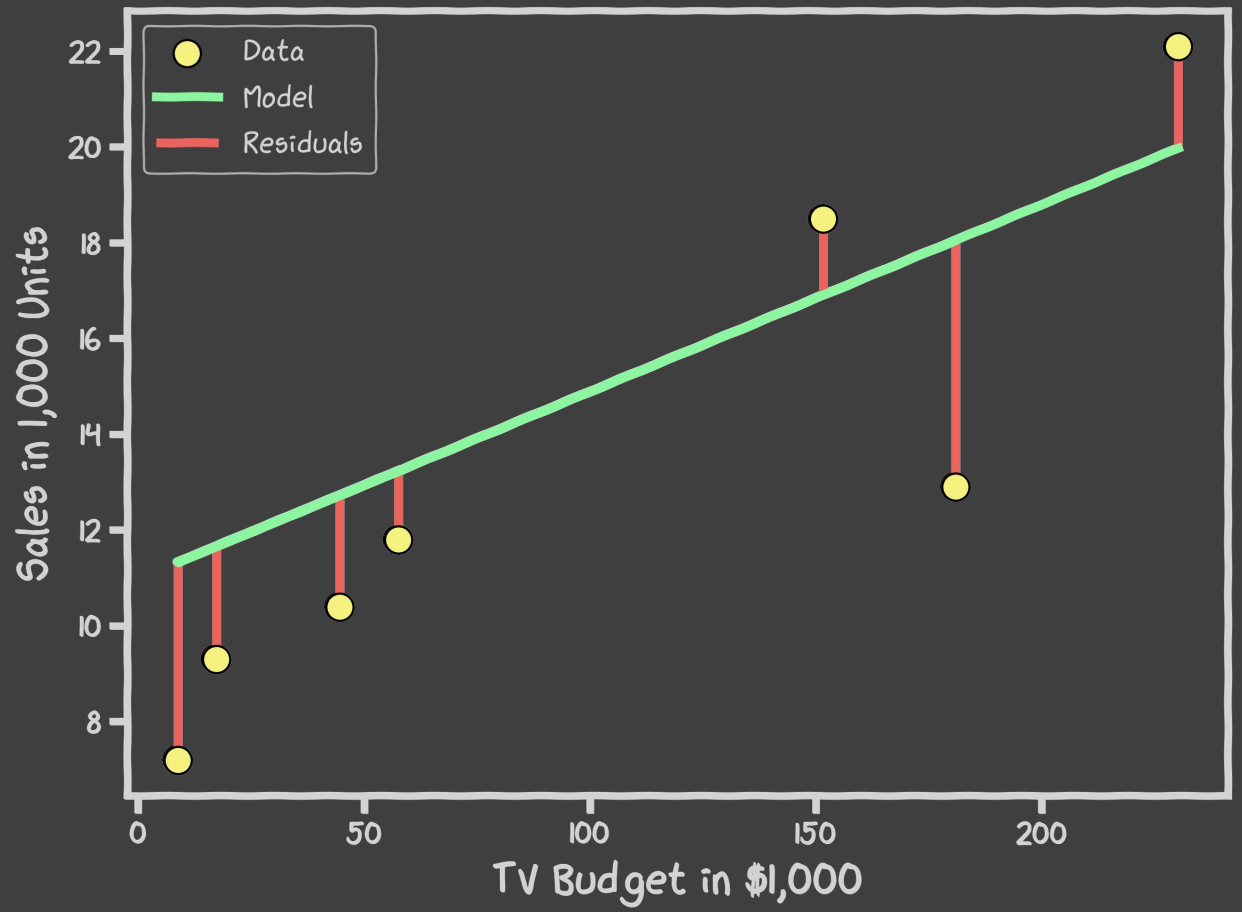
Estimate of the regression coefficients (cont)

Or this one?



Estimate of the regression coefficients (cont.)

Question: Which line is the best?



As before, for each observation (x_n, y_n), the absolute residuals, $r_i = |y_i - \hat{y}_i|$ quantify the error at each observation.

Estimate of the regression coefficients (cont.)

AGAIN, we use the MSE as our **loss function**,

$$L(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

We choose β_1 and β_0 that minimizes the predictive errors made by our model, i.e., minimize our loss function.

Then the optimal values, $\hat{\beta}_0$ and $\hat{\beta}_1$, should be:

$$\hat{\beta}_0, \hat{\beta}_1 = \underset{\beta_0,\beta_1}{\operatorname{argmin}} L(\beta_0, \beta_1).$$

WE CALL THIS **FITTING** OR **TRAINING** THE MODEL

Introducing...

sklearn.linear_model.LinearRegression

Methods

fit(X, y[, sample_weight])	Fit linear model.
get_params([deep])	Get parameters for this estimator.
predict(X)	Predict using the linear model.
score(X, y[, sample_weight])	Return the coefficient of determination R^2 of the prediction.

```
>>> from sklearn.linear_model import LinearRegression
>>> reg = LinearRegression()
>>> reg.fit(X, y)
>>> reg.coef_
array([1.2])
>>> reg.intercept_
3.2
>>> reg.predict(np.array([[3]]))
array([16.])
```

RECAP: Exercise

```
>>> from sklearn.linear_model import LinearRegression
>>> df = pd.read_csv('Advertising.csv')
>>> X= df[['TV']].values
>>> y = df['Sales'].values
```

RECAP: Exercise

```
>>> from sklearn.linear_model import LinearRegression
>>> df = pd.read_csv('Advertising.csv')
>>> X= df[['TV']].values
>>> y = df['Sales'].values
>>> reg = LinearRegression()
>>> reg.fit(X, y)
```

RECAP: Exercise

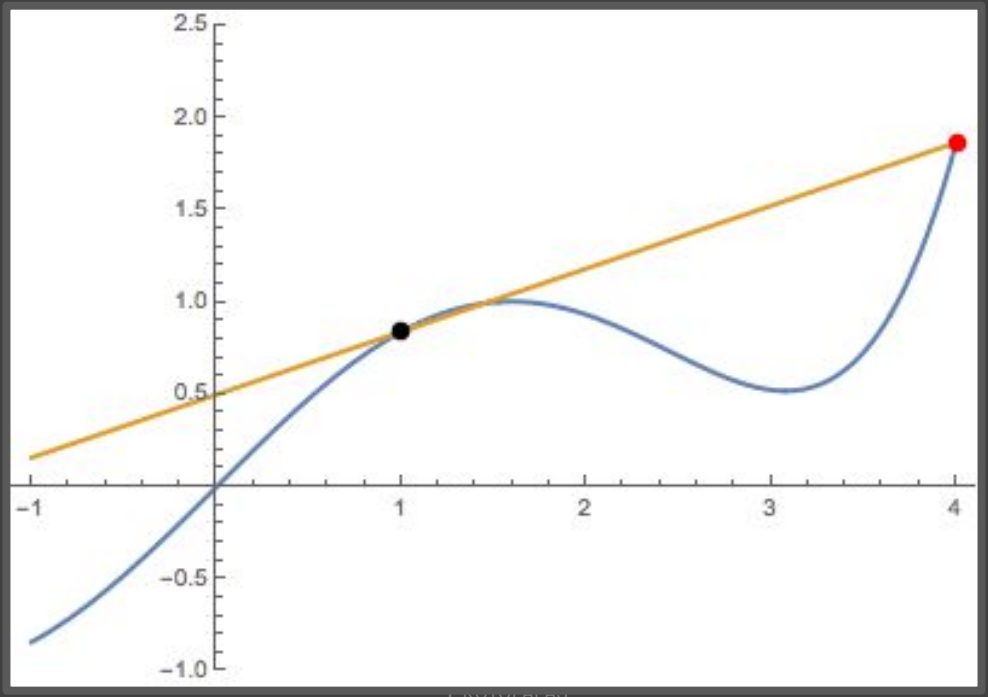
```
>>> from sklearn.linear_model import LinearRegression
>>> df = pd.read_csv('Advertising.csv')
>>> X= df[['TV']].values
>>> y = df['Sales'].values
>>> reg = LinearRegression()
>>> reg.fit(X, y)
>>> reg.coef_
array([[0.04665056]])
>>> reg.intercept_
array([7.08543108])
>>> reg.predict(np.array([[100]]))
array([[11.75048733]])
```

```
>>> reg.fit(X, y)
```

Derivative definition

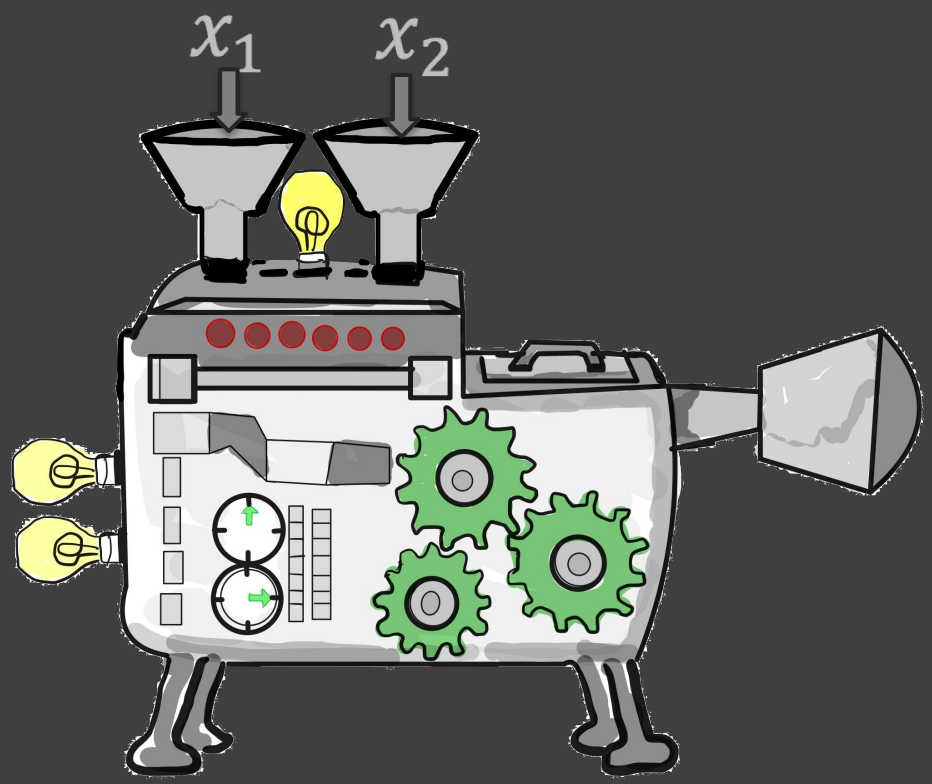
A derivative is the instantaneous rate of change of a single valued function. Given a function f(x) the derivative can be defined as:

$$f'(x) = \frac{df}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$



Partial derivatives

For a function f, the partial derivative is written as: $\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, ..$



$f(x_1, x_2)$

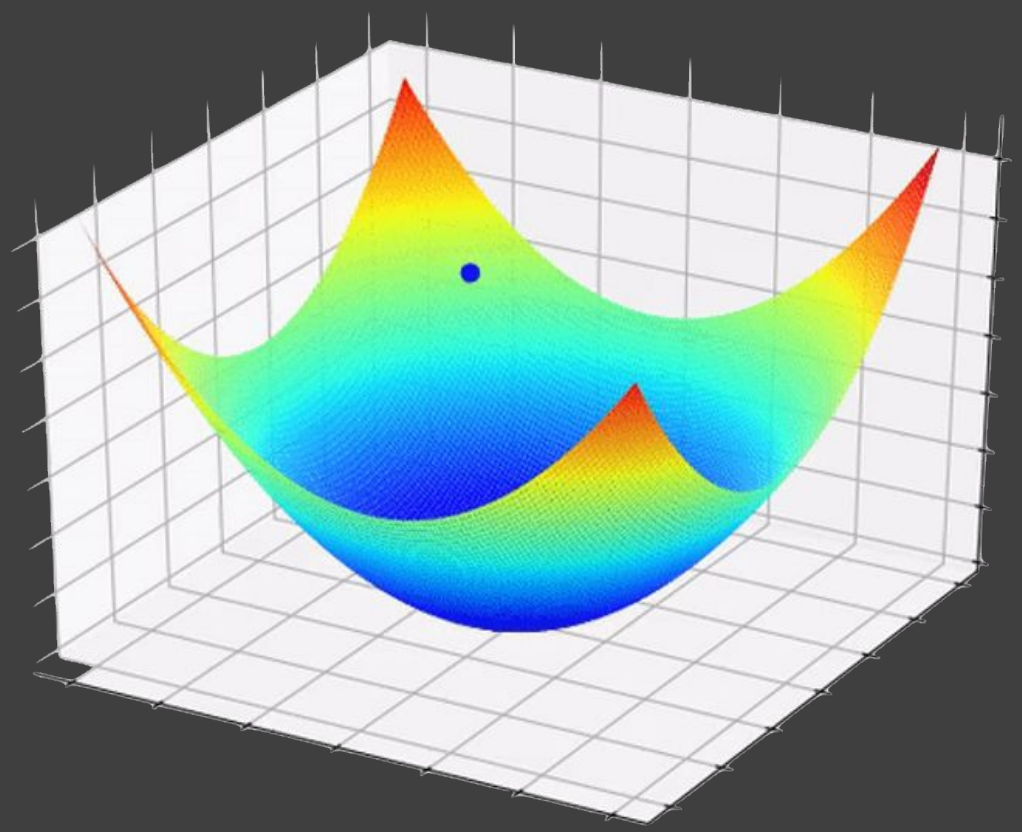
$f(x_1, x_2)$

$\frac{\partial f}{\partial x_1}$ $\frac{\partial f}{\partial x_2}$

What is the rate of change of the function with respect to one variable with the others held fixed?

Optimization

How does one minimize a loss function?



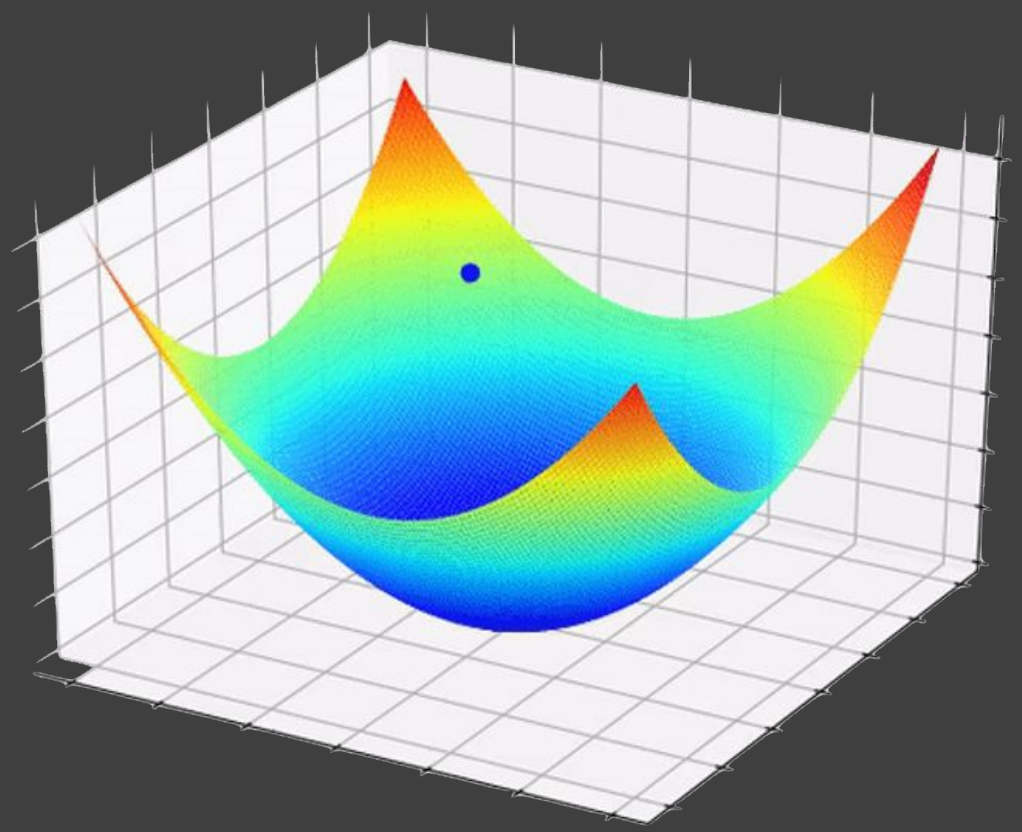
The global minima or maxima of $L(\beta_0, \beta_1)$ must occur at a point where the gradient (slope)

$$\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right] = 0$$

- Brute Force: Try every combination
- Exact: Solve the above equation
- Greedy Algorithm: Gradient Descent

Optimization

How does one minimize a loss function?



The global minima or maxima of $L(\beta_0, \beta_1)$ must occur at a point where the gradient (slope)

$$\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right] = 0$$

- Brute Force: Try every combination
- **Exact: Solve the above equation**
- Greedy Algorithm: Gradient Descent

Partial derivative example

If $L(\beta_0, \beta_1) = \left(y - (\beta_1 x + \beta_0)\right)^2$ then what is. $\frac{\partial l}{\partial \beta}$?

Looks like we're going to need the chain rule, but what is it? I forget

Partial derivative example

If $L(\beta_0, \beta_1) = \left(y - (\beta_1 x + \beta_0)\right)^2$ then what is $\frac{\partial l}{\partial \beta}$?

$$\frac{\partial L(f(\beta_0))}{\partial \beta_0} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial \beta_0}$$

Partial derivative $\frac{\partial L}{\partial \beta_0}$

If $L(\beta_0, \beta_1) = \left(y - (\beta_1 x + \beta_0)\right)^2$ then what is $\frac{\partial l}{\partial \beta}$?

$$L = (\, y - \beta_1 x - \beta_0 \,)^2$$

f

$$\frac{\partial L}{\partial \beta_0} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial \beta_0} \qquad L = f^2 \qquad \frac{\partial L}{\partial f} = 2f \qquad f = y - \beta_1 x - \beta_0 \qquad \frac{\partial f}{\partial \beta_0} = -1$$

$$\frac{\partial L}{\partial \beta_0} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial \beta_0} = -2f = -2(y - \beta_1 x - \beta_0)$$

Partial derivative $\frac{\partial L}{\partial \beta_1}$

If $L(\beta_0, \beta_1) = \left(y - (\beta_1 x + \beta_0)\right)^2$ then what is $\frac{\partial l}{\partial \beta}$?

$$L = (y - \beta_1 x - \beta_0)^2$$

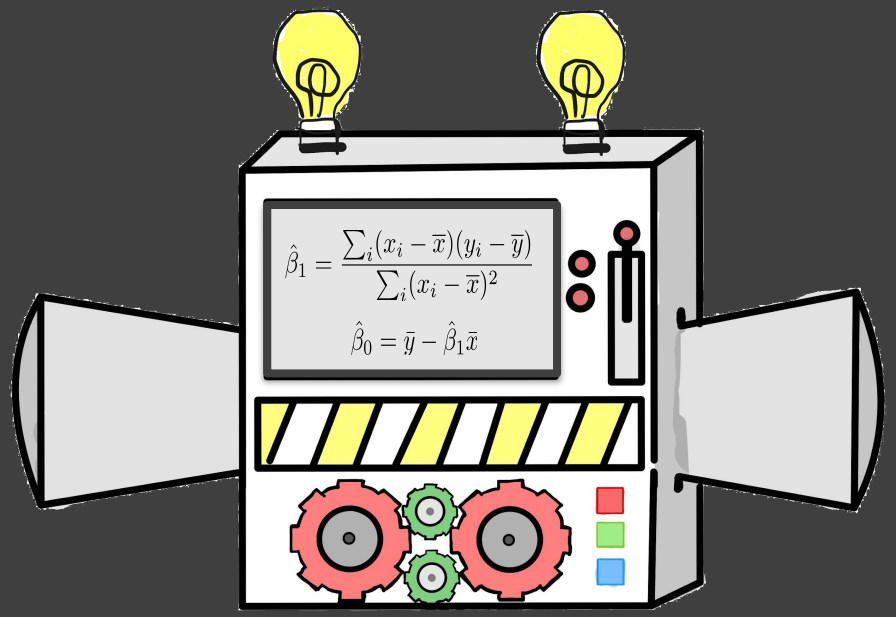
$$\frac{\partial L}{\partial \beta_1} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial \beta_1} \qquad L = f^2 \qquad \frac{\partial L}{\partial f} = 2f \qquad f = y - \beta_1 x - \beta_0 \qquad \frac{\partial f}{\partial \beta_1} = -x$$

$$\frac{\partial L}{\partial \beta_1} = \frac{\partial L}{\partial f}\frac{\partial f}{\partial \beta_1} = -2xf = -2x(y - \beta_1 x - \beta_0)$$

Optimization

$$\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right]=0$$

$$\frac{\partial L}{\partial \beta_0} = -2(y - \beta_1 x - \beta_0) = 0$$



$$\frac{\partial L}{\partial \beta_1} = -2x(y - \beta_1 x - \beta_0)=0$$

Optimization

$$\hat{\beta}_1 = \frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sum_i (x_i - \overline{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

Summary: Estimate of the regression coefficients

We use MSE as our **loss function**,

$$L(\beta_0, \beta_1) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{y}_i)^2 = \frac{1}{n}\sum_{i=1}^{n}[y_i - (\beta_1 x_i + \beta_0)]^2$$

We choose β_1 and β_0 in order to minimize the predictive errors made by our model, i.e. minimize our loss function.

Then the optimal values for $\hat{\beta}_0$ and $\hat{\beta}_1$ should be:

$$\widehat{\beta}_0, \widehat{\beta}_1 = \underset{\beta_0,\beta_1}{\operatorname{argmin}} L(\beta_0, \beta_1).$$

WE CALL THIS **FITTING** OR **TRAINING** THE MODEL

Estimate of the regression coefficients: analytical solution

Take the gradient of the loss function and find t e the gradient is zero: $\nabla L = \left[\frac{\partial L}{\partial \beta_0}, \frac{\partial L}{\partial \beta_1}\right]$=0

Finding the exact solution only works for rare cases. Linear regression is one of such rare cases.

$$\hat{\beta}_1 = \frac{\sum_i (x_i - \overline{x})(y_i - \overline{y})}{\sum_i (x_i - \overline{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

where $\bar{y}$ and $\bar{x}$ are sample means.

The line:

is called the **regression line**.

$$\widehat{Y} = \widehat{\beta}_1 X + \widehat{\beta}_0$$