Do not distribute course material

You may not and may not allow others to reproduce or distribute lecture notes and course materials publicly whether or not a fee is charged.

Topic 1
Subcategory of Supervised Learning
Linear Model
Simple Linear Regression

A model assuming a linear relationship between the input variables and the output variable

Studied for over 200 years

Simple = only one input variable

CS/EE-UY 4563: INTRODUCTION TO MACHINE LEARNING

PROF. LINDA SELLIE

Some of the slides are from Prof. Sundeep Rangan

Some approaches are taken from CMU 18-661 Introduction to Machine Learning

Learning Objectives

- ❑ How to load data from a text file
- ❑ How to visualize data via a scatter plot
- ❑ Describe a linear model for data
  - Identify the label (target variable) and feature (predictor)
- ❑ Understand the objective function for linear regression
- ❑ Understand how partial derivatives can be used in a convex optimization problem
- ❑ Optimization:
  - Compute optimal parameters for the model using a closed form solution
  - Compute the optimal parameters for the model using gradient descent
- ❑ Evaluation:
  - Able to compute R^2
  - Able to visually determine goodness of fit and identify different causes for poor fit

Regression means real valued output

Used in statistics and Economics

THE TERM REGRESSION COMES FROM STATISTICS - WHENEVER HAVE A REAL VALUED OUTPUT WE CALL IT REGRESSION

Notation

We will use the notation (mostly…) from Stanford and the Deep Learning Book

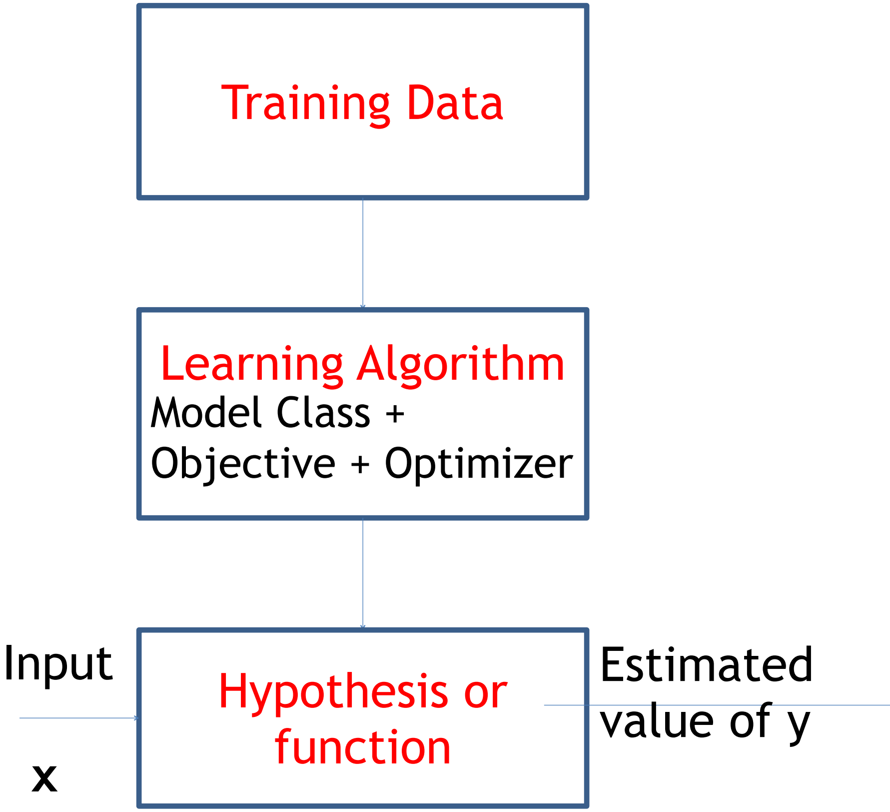
- **Input (features):** $\mathbf{x} \in \mathbb{R}^d$ $(\mathbf{x}^{(i)}$ for the ith example
- **Output (target/label):** $y \in \mathbb{R}$ $(y^{(i)}$ for the ith example)
- **Training data:** $(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)}),$

$$X = \begin{bmatrix} \mathbf{x}^{(1)T} \\ \mathbf{x}^{(2)T} \\ \vdots \\ \mathbf{x}^{(N)T} \end{bmatrix} \text{ design matrix}$$

- The number of training examples: N
- **Model class (hypothesis class):** $h : \mathbf{x} \rightarrow y$, with $h(\mathbf{x}) = w_0 + \sum_{i=1}^{d} w_i x_i$

w_0, w_1 are the *weights/parameters*

w_0 is the *bias*



Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

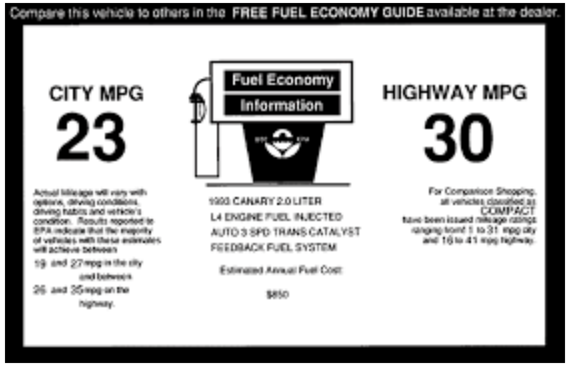
Predicting Trends

- Example: Predicting mpg for a car

mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
?	8	383.0	170.0	3563.0	10.0	70	1	dodge challenger se

Example: What Determines mpg in a Car?

- What engine characteristics determine fuel efficiency?
- Why would a data scientist be hired to answer this question?
  - Not to help purchasing a specific car
    - The mpg for a currently available car is already known.
    - (If the car company isn't lying?)
  - To guide building new cars
    - Understand what is reasonably achievable before full design
  - To find cars that are outside the trend
    - Example: What cars give great mpg for the cost or size?



Choose the average MPG?

Keeping it simple

- Lets assume that *one* of the features can approximately predict the MPG

Lets plot the data

Demo on NYU Classes

Simple Linear Regression for Automobile mpg Data

Loading the Data

The python `pandas` library is a powerful package for data analysis. In this course, we will use a small portion of its features -- just reading and writing data from files. After reading the data, we will convert it to `numpy` for all numerical processing including running machine learning algorithms.

We begin by loading the packages.

In [86]:
```
import pandas as pd
import numpy as np
```

The data for this demo comes from a survey of cars to determine the relation of mpg to engine characteristics. The data can be found in the UCI library: https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg

You can directly read the data in the file, https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data We will load the data into ipython notebook, using the pandas library. Unfortunately, the file header does not include the names of the fields,

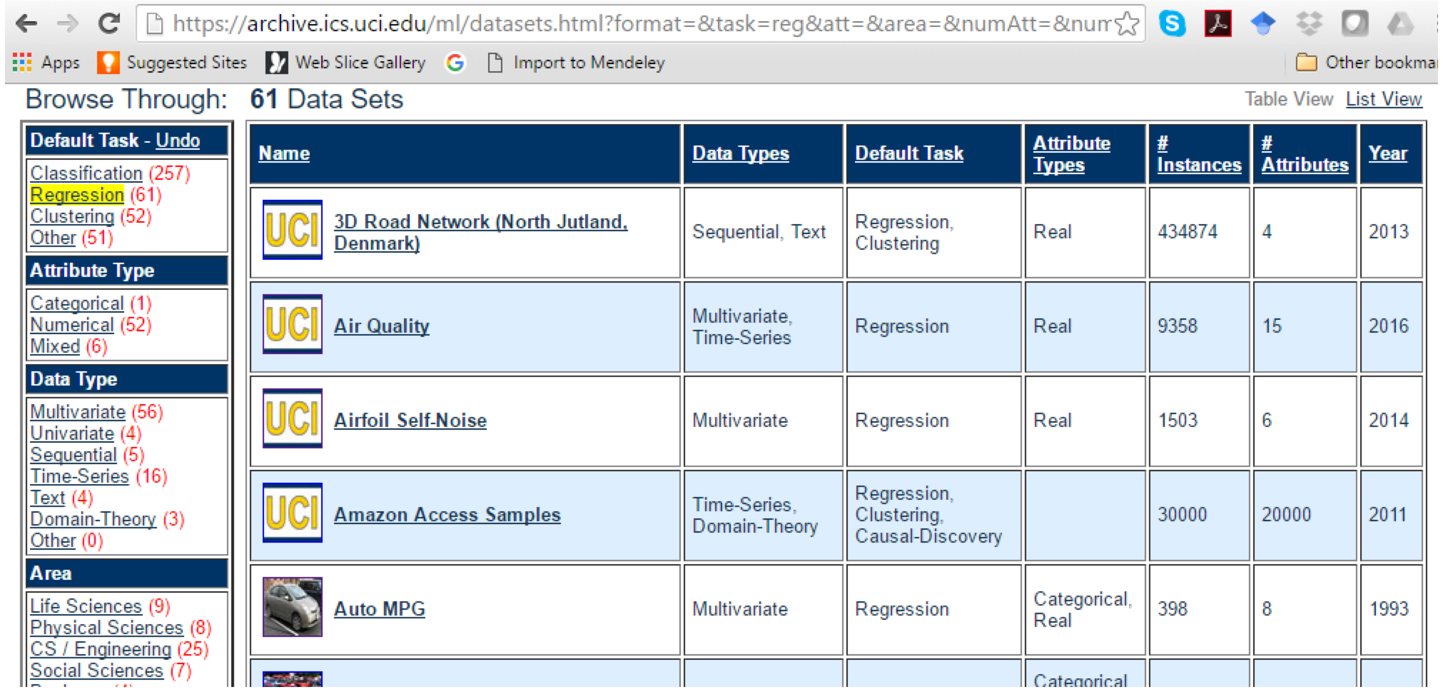
Pandas make it easy to work with two dimensional tables

The posted demo shows how to:

- Load data from a text file using the `pandas` package
- Create a scatter plot of data
- Handle missing data
- Fit a simple linear model
- Plot the linear fit with the test data
- Use a nonlinear transformation for an improved fit

Getting Data

- Data from UCI dataset library: [UCI Machine Learning Repository]()



https://archive.ics.uci.edu/ml/datasets.html?format=&task=reg&att=&area=&numAtt=&num

Apps Suggested Sites Web Slice Gallery Import to Mendeley Other bookma

Browse Through: **61** Data Sets

Table View List View

Default Task - Undo
Classification (257)
Regression (61)
Clustering (52)
Other (51)

Attribute Type
Categorical (1)
Numerical (52)
Mixed (6)

Data Type
Multivariate (56)
Univariate (4)
Sequential (5)
Time-Series (16)
Text (4)
Domain-Theory (3)
Other (0)

Area
Life Sciences (9)
Physical Sciences (8)
CS / Engineering (25)
Social Sciences (7)

Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes	Year
3D Road Network (North Jutland, Denmark)	Sequential, Text	Regression, Clustering	Real	434874	4	2013
Air Quality	Multivariate, Time-Series	Regression	Real	9358	15	2016
Airfoil Self-Noise	Multivariate	Regression	Real	1503	6	2014
Amazon Access Samples	Time-Series, Domain-Theory	Regression, Clustering, Causal-Discovery		30000	20000	2011
Auto MPG	Multivariate	Regression	Categorical, Real	398	8	1993
			Categorical			

Today’s lecture

Loading the Data in Jupyter Notebook
Try 1: The Wrong Way!

```
import pandas as pd
import numpy as np
```

```
In [67]: names = ['mpg', 'cylinders','displacement', 'horsepower',
                  'weight', 'acceleration', 'model year', 'origin', 'car name']
```

```
In [122]: df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data')
```

```
In [123]: df.head(6)
```

Out[123]:

	18.0 8 307.0 130.0 3504. 12.0 70 1 "chevrolet chevelle malibu"
0	15.0 8 350.0 165.0 3693. 11...
1	18.0 8 318.0 150.0 3436. 11...
2	16.0 8 304.0 150.0 3433. 12...
3	17.0 8 302.0 140.0 3449. 10...
4	15.0 8 429.0 198.0 4341. 10...
5	14.0 8 454.0 220.0 4354. 9...

- Python pandas library
  - pd.read_csv command
  - Reads URL or file location
- Creates a dataframe object
  - http://pandas.pydata.org/pandas-docs/stable/dsintro.html#dataframe
- Problems:
  - Did not parse columns
    - All data in a single column
    - Read_csv assumes columns are delimited by commas
  - Mistakes first line as header

Loading the Data in Jupyter
Try 2: Fixing the Errors

In [125]:
```
df = pd.read_csv('https://archive.ics.uci.edu/ml/machine-learning-databases/'+
                 'auto-mpg/auto-mpg.data',
                 header=None,delim_whitespace=True,names=names,na_values='?')
```

You can display a first few lines of the dataframe by using head command:

In [126]:
```
df.head(6)
```

Out[126]:

	mpg	cylinders	displacement	horsepower	weight	acceleration	model year	origin	car name
0	18	8	307	130	3504	12.0	70	1	chevrolet chevelle malibu
1	15	8	350	165	3693	11.5	70	1	buick skylark 320
2	18	8	318	150	3436	11.0	70	1	plymouth satellite
3	16	8	304	150	3433	12.0	70	1	amc rebel sst
4	17	8	302	140	3449	10.5	70	1	ford torino
5	15	8	429	198	4341	10.0	70	1	ford galaxie 500

- Fix the arguments in read_csv
- pd.read_csv has many options to deal with a wide variety of differently formatted data sets
- When you get a problem:
  - **Google is your friend!**
  - You are not the first to have these problems
  - Official documentation exists
- Ask questions on Brightspace if you get stuck

Visualizing the Data using a scatter plot

```
In [150]: xstr = 'horsepower'
          X = np.array(df[xstr])
          y = np.array(df['mpg'])
```

```
In [146]: import matplotlib
          import matplotlib.pyplot as plt
          %matplotlib inline
```

```
In [151]: plt.plot(X,y,'o')
          plt.xlabel(xstr)
          plt.ylabel('mpg')
          plt.grid(True)
```

- We will plot data in Python using Matplotlib
- A nice tutorial: https://matplotlib.org/users/pyplot_tutorial.html
- How could you predict the mpg of a car not in the data as a function of the horsepower? Pair share

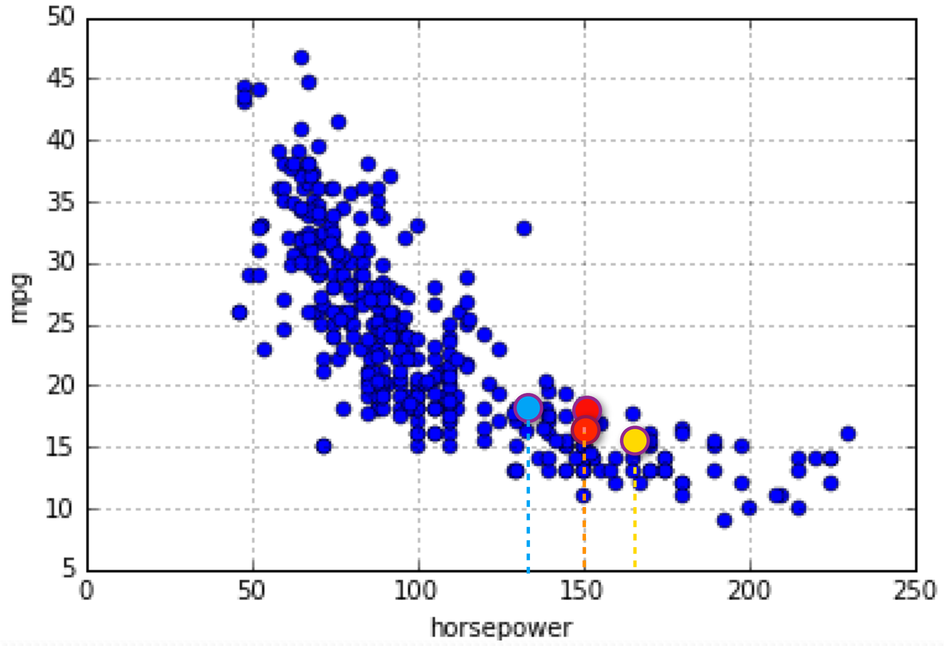
$(\mathbf{x}^{(1)} = 130,\ y^{(1)} = 18)$

$(\mathbf{x}^{(2)} = 165,\ y^{(2)} = 15)$

$(\mathbf{x}^{(3)} = 150,\ y^{(3)} = 18)$

$(\mathbf{x}^{(4)} = 150,\ y^{(4)} = 17)$

...



$\mathbf{x}^{(1)} = 130 \quad \mathbf{x}^{(2)} = 165$

$\mathbf{x}^{(3)} = 150$

$$X = \begin{bmatrix} 130 \\ 165 \\ 150 \\ 150 \\ \dots \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ \dots \end{bmatrix}$$

Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

Approach

≈ approx. equal

Warning! Stanford notes use θ instead of w for the parameters

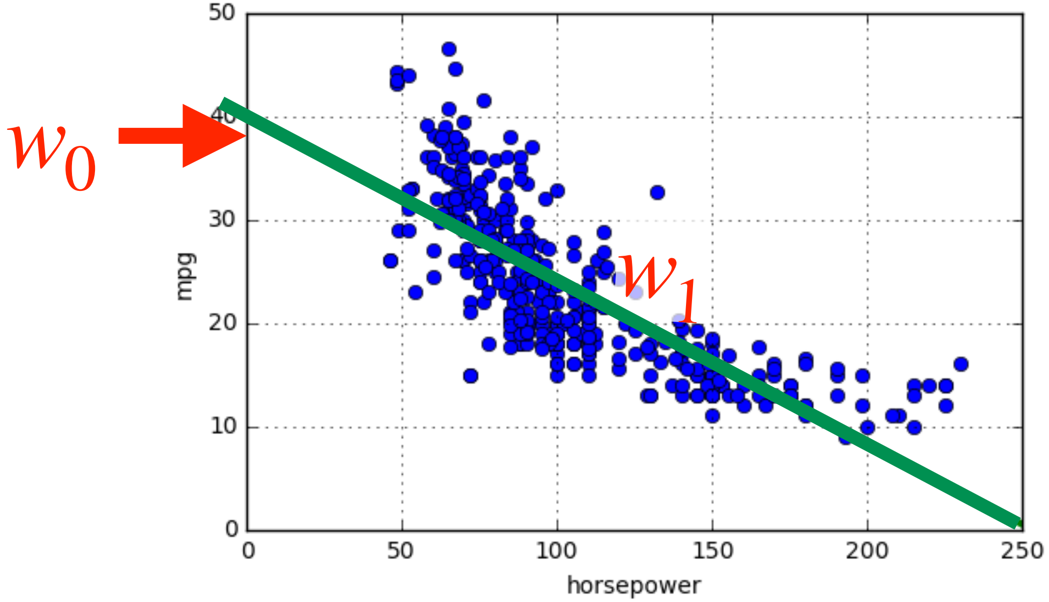
$y \approx w_0 + w_1\mathbf{x}$

MPG $\approx w_0 + w_1$ Horsepower

Learn parameters w_0, w_1 of the green line

Car 1: $w_0 + w_1 \cdot 130 = 18$

Car 2: $w_0 + w_1 \cdot 165 = 15$



We are changing the notation from what you are used to seeing when we describe a line. Instead of $mx + b = y$, we are using $w_0 + w_1\mathbf{x}$

Approach

$y \approx w_0 + w_1 \mathbf{x}$

MPG $\approx w_0 + w_1$ Horsepower

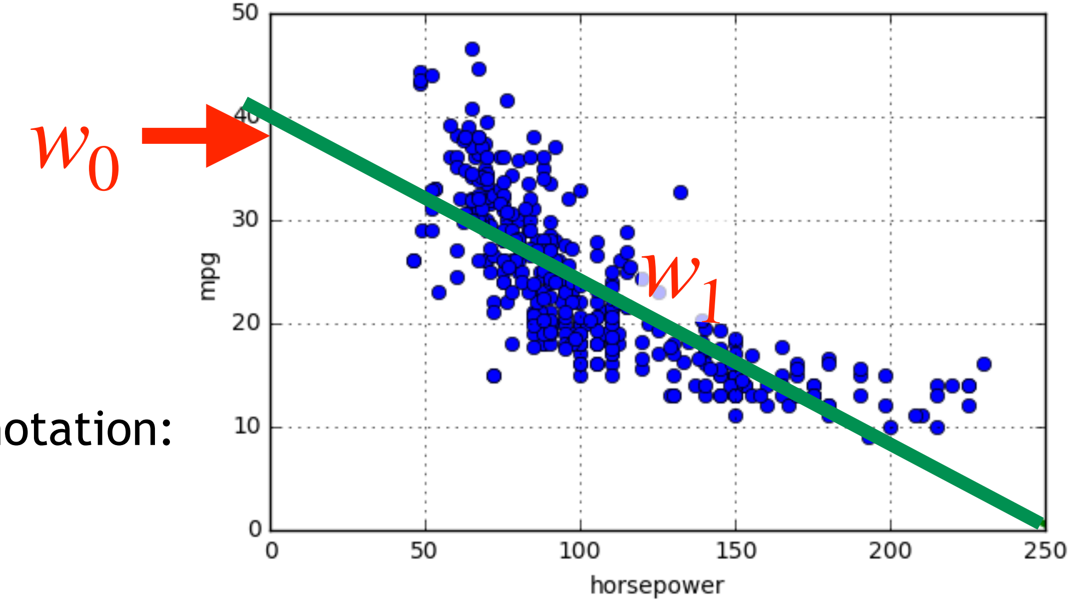
Learn parameters w_0, w_1 of the green line

Car 1: $w_0 + w_1 \cdot 130 = 18$

Car 2: $w_0 + w_1 \cdot 165 = 15$

We can compactly represent this in matrix notation:

$$\begin{bmatrix} 1 & 130 \\ 1 & 165 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 18 \\ 15 \end{bmatrix}$$



Finding w_0, w_1 if we only use two examples

$$\begin{bmatrix} 1 & 130 \\ 1 & 165 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \begin{bmatrix} 18 \\ 15 \end{bmatrix}$$

$$\begin{bmatrix} w_0 \\ w_1 \end{bmatrix} = \left(\begin{bmatrix} 1 & 130 \\ 1 & 165 \end{bmatrix} \right)^{-1} \begin{bmatrix} 18 \\ 15 \end{bmatrix}$$

$$= \begin{bmatrix} 4.71 & -3.71 \\ -.03 & .03 \end{bmatrix} \begin{bmatrix} 18 \\ 15 \end{bmatrix}$$

$$= \begin{bmatrix} 29.14 \\ -0.09 \end{bmatrix}$$

If we use more data, how can we estimate w_0, w_1?

horsepower	mpg
130.0	18.0
165.0	15.0
150.0	18.0
150.0	16.0
...	...

Problem! There isn't a w_0, w_1 that will satisfy all the equations

Want to predict the best MPG based on the horsepower

MPG = $w_0 + w_1$horsepower + unexplainable_stuff

horsepower	mpg
130.0	18.0
165.0	15.0
150.0	18.0
150.0	16.0
...	...

Prediction car 1: $w_0 + w_1 \cdot 130$

Prediction car 2: $w_0 + w_1 \cdot 165$

Prediction car 3: $w_0 + w_1 \cdot 150$

Prediction car 4: $w_0 + w_1 \cdot 150$

...

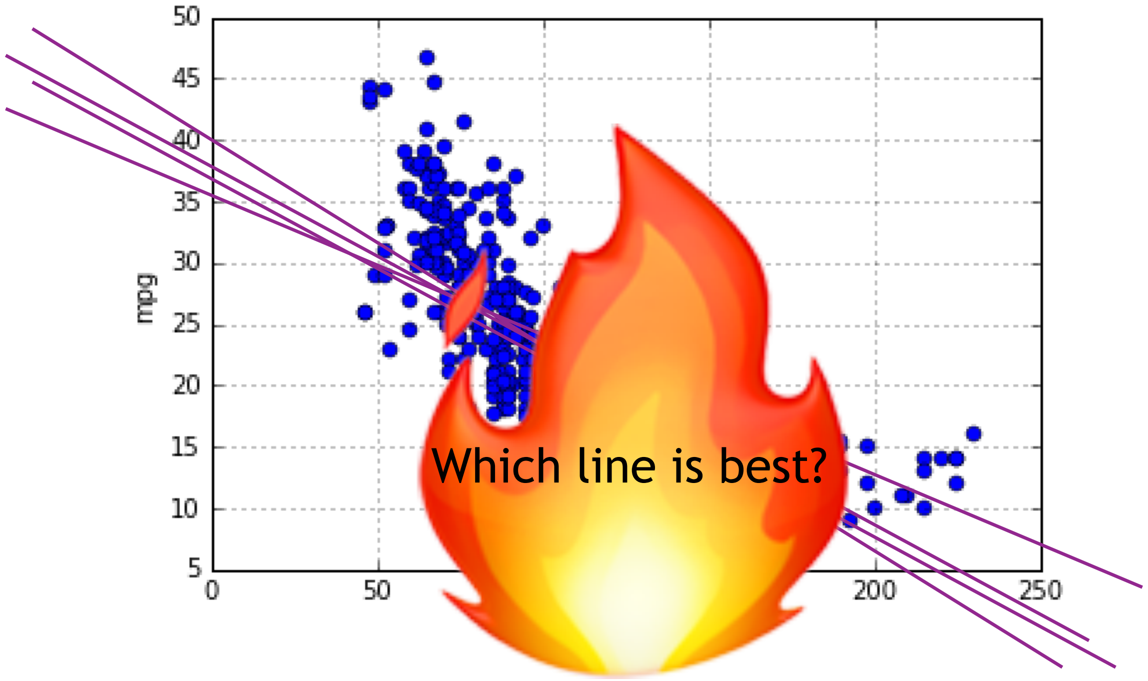
Learn w_0 and w_1

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful." George Box

Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

Linear Model



Pair share

How do we decide which line is ‘best’?

What does it mean for one line to be better than another line?

HOW MUCH DOES A POINT NOT ON THE LINE **COST**?

WE COULD CREATE A **COST FUNCTION**. THE BEST LINE HAS THE LOWEST COST WHEN SUMMED OVER ALL THE EXAMPLES

WHAT **COST FUNCTION** SHOULD WE USE?

Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

We choose to minimize the variation around the line

How to measure errors?

mpg ≈ w_0 + w_1 horsepower

$\hat{y} = w_0 + w_1\mathbf{x}$

If $w_0 = 39.94, w_1 = -0.16$

$\hat{y} = h(\mathbf{x}) = 39.94 + (-0.16)\mathbf{x}$

horsepower	mpg
130.0	18.0
165.0	15.0
150.0	18.0
150.0	16.0
...	...

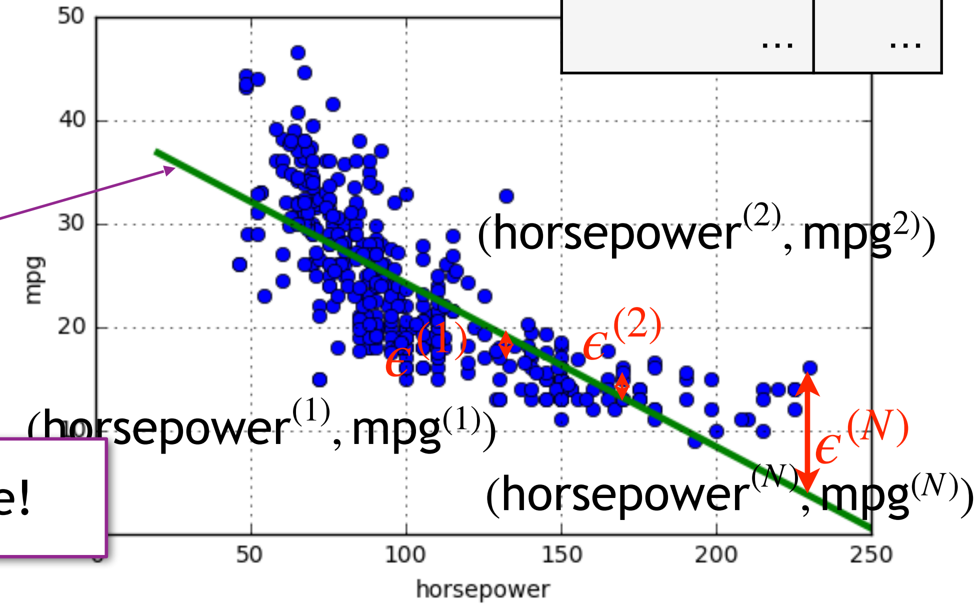
- Absolute value of difference?

$|y - \hat{y}|$ = |MPG − predicted MPG|

$$
\begin{aligned}
&|18 - (w_0 + w_1 * 130)| \\
+&|15 - (w_0 + w_1 * 165)| \\
+&|18 - (w_0 + w_1 * 150)| \\
+&|16 - (w_0 + w_1 * 150)| \\
&\ldots
\end{aligned}
$$

Note that
$|y - \hat{y}| = |\hat{y} - y|$
E.g. $|w_0 + w_1 * 130 - 18|$
$= |18 - (w_0 + w_1 * 130)|$

Regression line



- Squared value of difference?

$(y - \hat{y})^2$ = (MPG − predicted MPG)^2

Differentiable!

$$
\begin{aligned}
&(18 - (w_0 + w_1 * 130))^2 \\
+&(15 - (w_0 + w_1 * 165))^2 \\
+&(18 - (w_0 + w_1 * 150))^2 \\
+&(16 - (w_0 + w_1 * 150))^2 \\
&\ldots
\end{aligned}
$$

Note that $(y - \hat{y})^2 = (\hat{y} - y)^2$
E.g. $(w_0 + w_1 * 130 - 18)^2$
$= \left(18 - (w_0 + w_1 * 130 - 18)\right)^2$

Least Squares Model Objective function

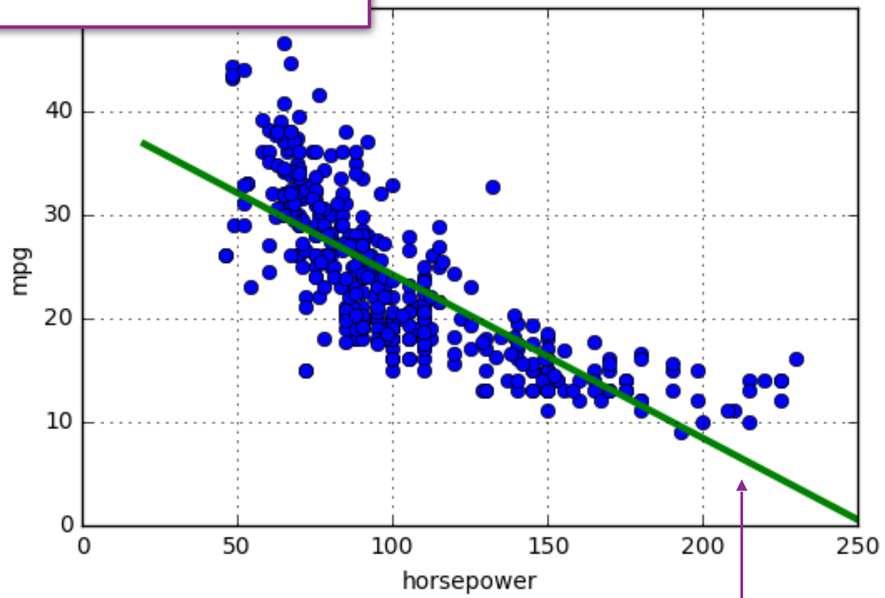
- **Model** relationship between horsepower and mpg as a line $\hat{y} = h(\mathbf{w}) = w_0 + w_1\mathbf{x}$
- **Objective function:** Find parameters $w=(w_0,w_1)^T$ to minimize cost

horsepower	mpg
130.0	18.0
165.0	15.0
150.0	18.0
150.0	16.0
...	...

$$\text{RSS}(w_0, w_1) = \sum_{i=1}^{N} \left(y^{(i)} - (w_0 + w_1\mathbf{x}^{(i)})\right)^2 = \sum_{i=1}^{N} \left(y^{(i)} - \hat{y}^{(i)}\right)^2$$

Residual Sum of Squares
Also called the sum of squared residuals (SSR) and sum of squared errors (SSE)

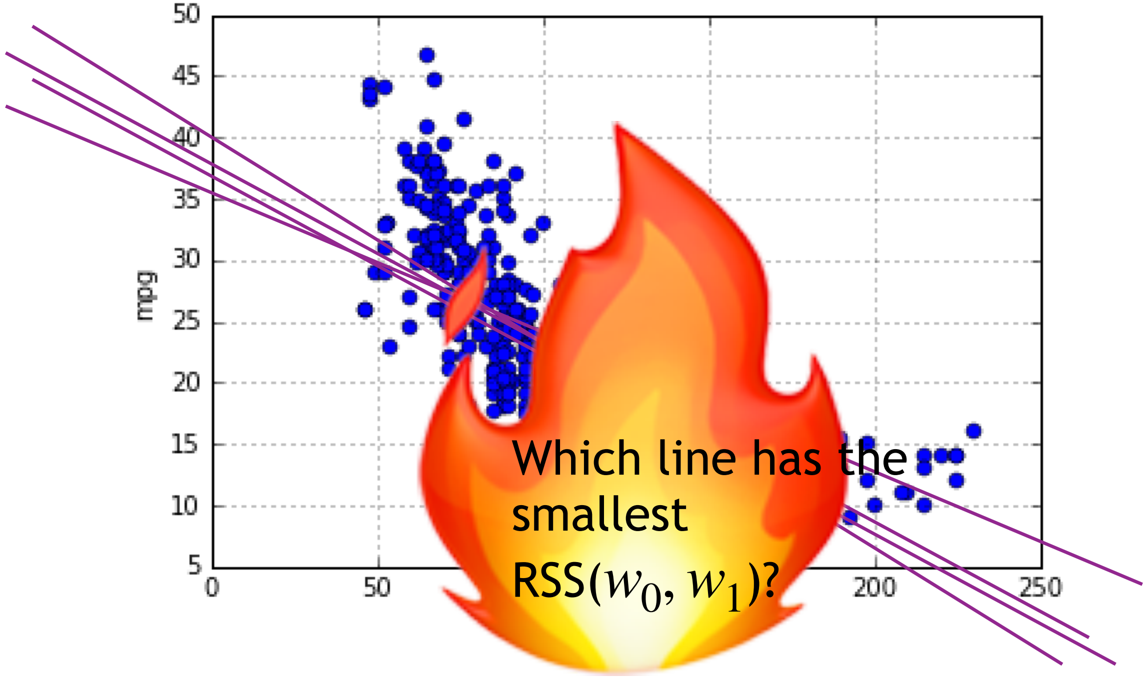
$$\text{MSE}(w_0, w_1) = \frac{1}{N}\text{RSS}(w_0, w_1) = E_{\text{in}}(w_0, w_1)$$

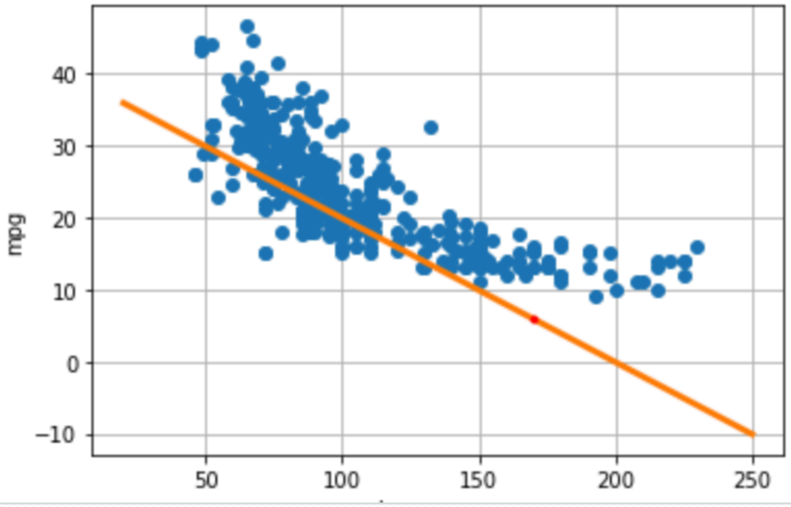


Regression line

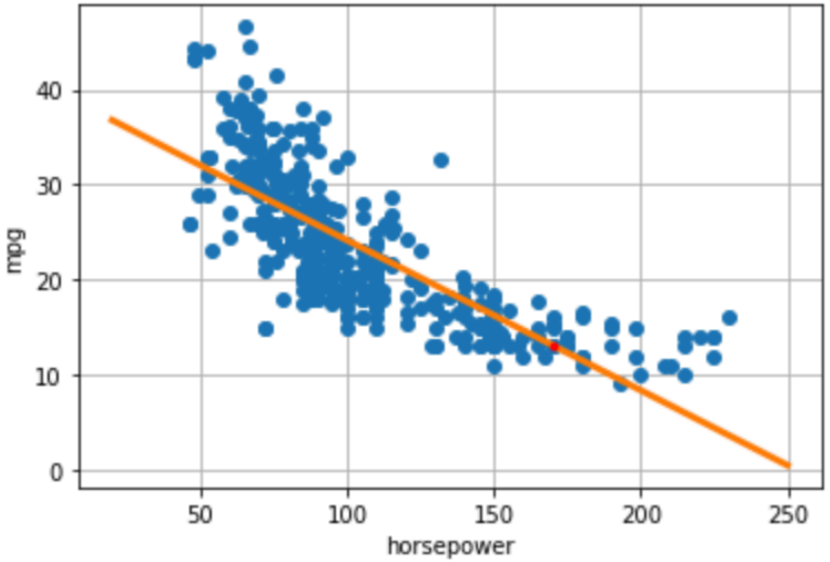
mpg = $w_0 + w_1$ horsepower

Linear Model



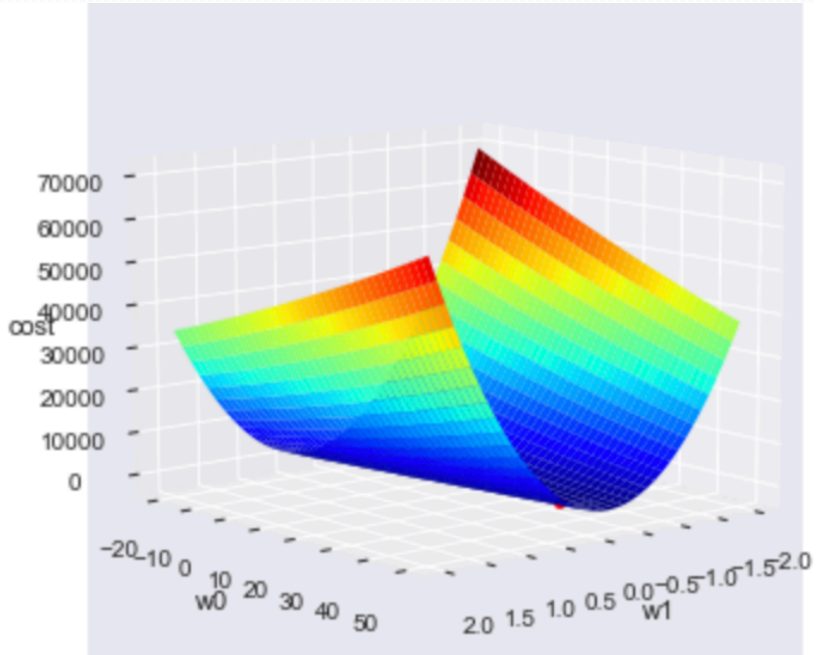


horsepower	mpg
130.0	18.0
165.0	15.0
150.0	18.0
150.0	16.0
…	…



$$RSS(39.9, \ -0.2)$$
$$= (39.9 + (-0.2) * 130 - 18)^2 + (39.9 + (-0.2) * 165 - 15)^2 + \cdots$$
$$= 18142.38$$

$$RSS(39.94, \ -0.16)$$
$$= (39.94 + (-0.16) * 130 - 18)^2 + (39.94 + (-0.16) * 165 - 15)^2 + \cdots$$
$$= 9385.92$$



Finding Parameters via Optimization
A general ML recipe

General ML problem	Simple linear regression
Get data	Data: $(\mathbf{x}^{(i)}, y^{(i)}), i = 1,\ldots,N$
Find a hypothesis class/model class with parameters	Linear model: $\hat{y} = w_0 + w_1\mathbf{x}$
Pick a loss function ◦ Measures goodness of fit model to data ◦ Function of the parameters	Loss function: $RSS(w) = \sum_{i=1}^{N} (y^{(i)} - \hat{y}^{(i)})^2$
Find parameters that minimizes loss	Select w= $\begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$ to minimize loss function

Remember
$$\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$$

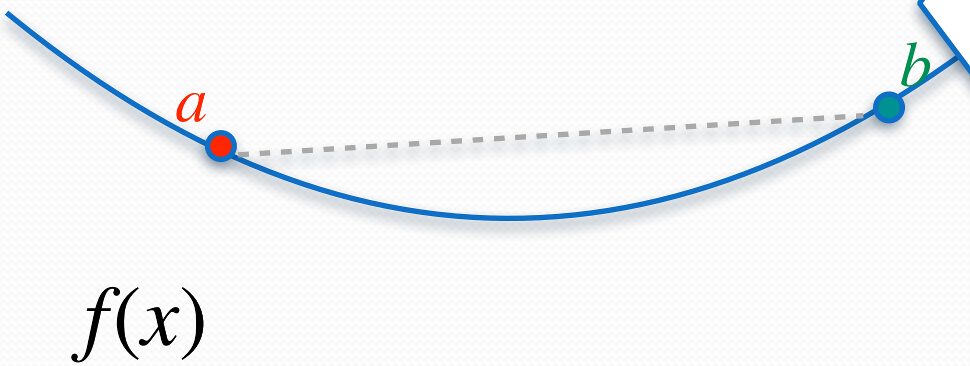
I.e. how do we find the line that ***minimizes*** the ***cost function?***

How do we find the *global minimum*?

$$\min_{\mathbf{w}} \mathrm{RSS}(\mathbf{w})$$

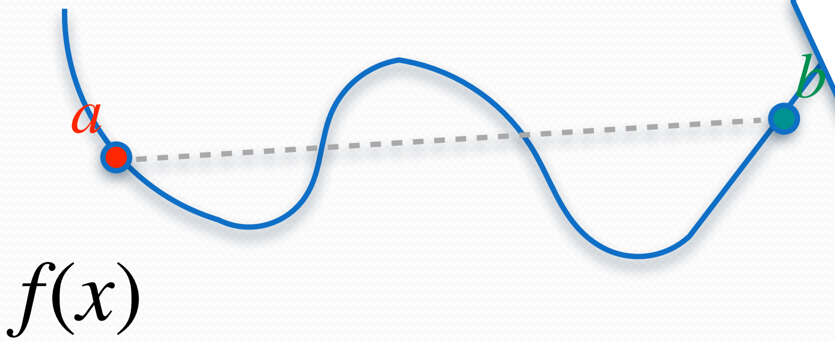
Which values for w_0, w_1 minimizes $\mathrm{RSS}(\mathbf{w})$?

Convex function

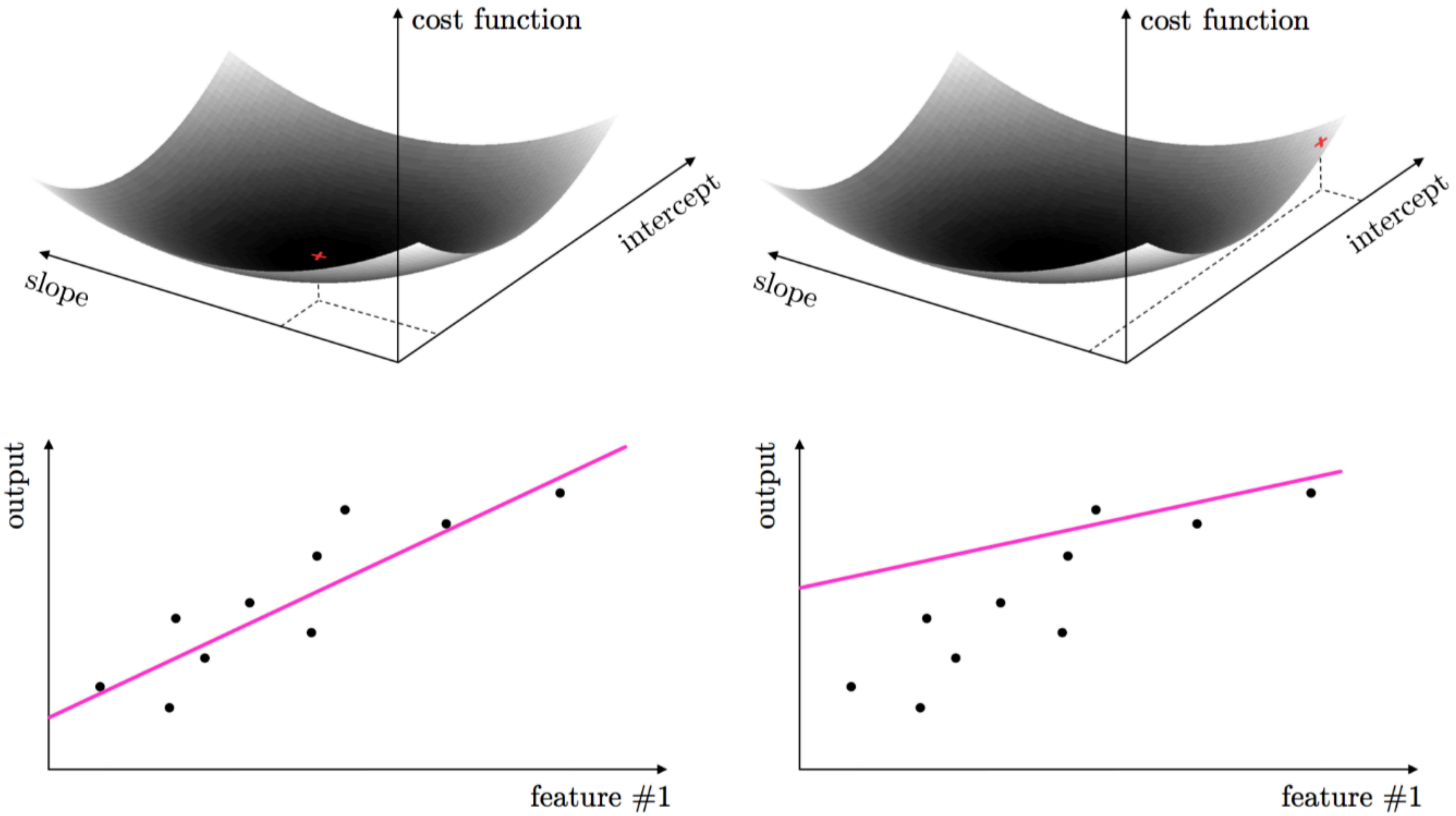


Only one global minimum value

Not a convex function



More than one local minimum value



Slide from Machine Learning Refined: Foundations, Algorithms, and Applications

…this would be even more difficult if we had more dimensions

I think we need an **algorithmic approach** to find the **optimal value**

Mathematical optimization

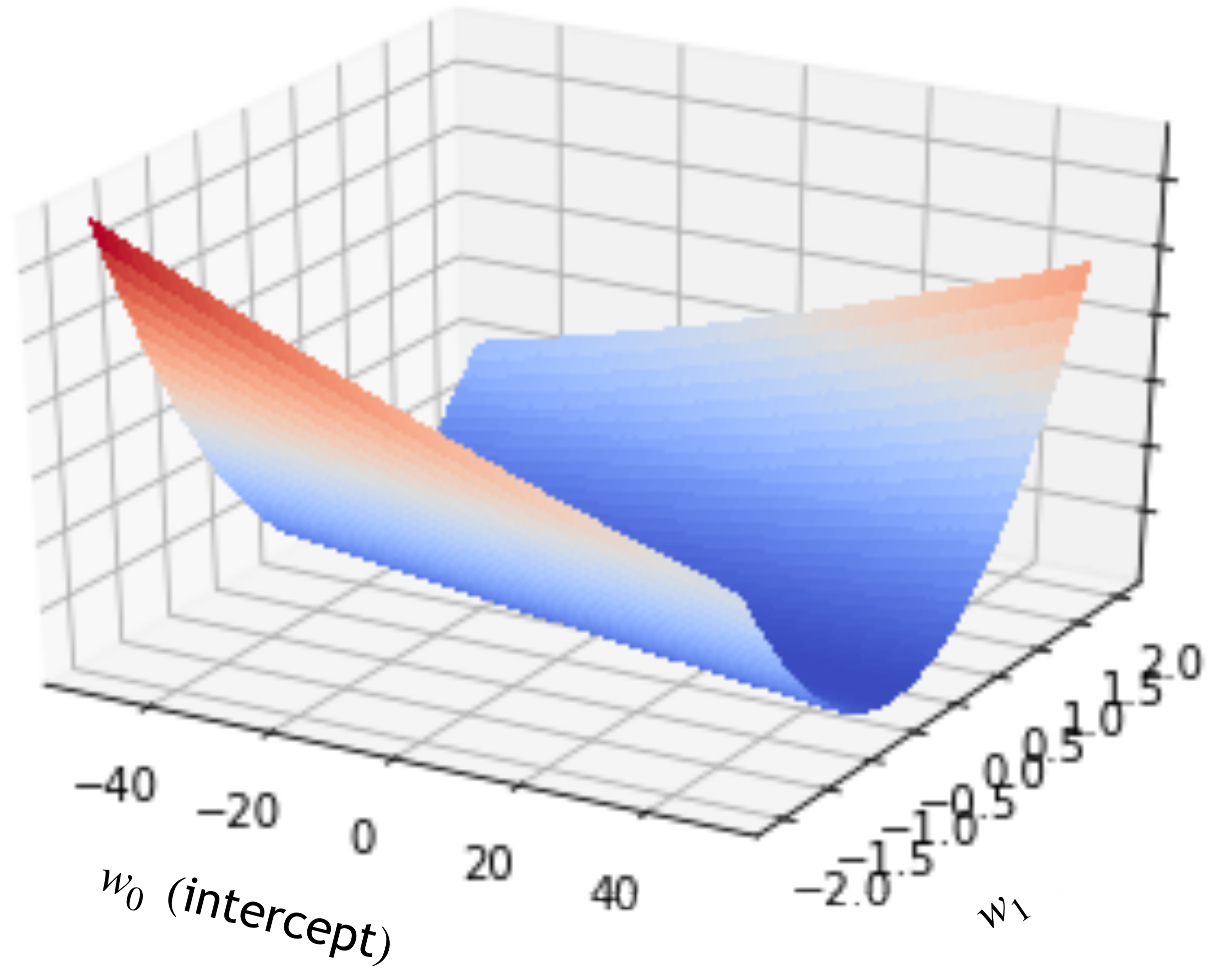
Pair share

Discuss various methods for finding the optimal parameters, i.e. values for $\mathbf{w} = \begin{bmatrix} w_0 \\ w_1 \end{bmatrix}$

Discussion of ideas

Global optimization methods

Local optimization methods



Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

Outline

- Motivating Example: Predicting the mpg of a car
- Model: Linear Model
- Objective function: Least Squares Fit Problem
- Global Op : LS Fit Solution
- Local Optimizer: Gradient Descent
- Assessing Goodness of Fit
- Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

Global optimization for Linear regression

Given a dataset $D = \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$ how do we find the optimal weight vector $\mathbf{w}$ to minimize

$$\text{RSS}(w_0, w_1) = \sum_{i=1}^{N} \left(y^{(i)} + (-w_0 + -w_1\mathbf{x}^{(i)})\right)^2$$

- We don't know the parameters w_0, w_1 (they are also called coefficients)
- We could look over all possible lines to find the optimum...
- We can derive the partial derivative and then set the derivative to **zero** to find the minimum value
- Notice that since the error function is convex (we cannot get trapped at a local minimum - the solution is unique)

$$\frac{\partial RSS}{\partial w_0} = \sum_{i=1}^{N} 2(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(-1) \quad = -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)}) = 0$$

$$\frac{\partial RSS}{dw_1} = \sum_{i=1}^{N} 2(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(-\mathbf{x}^{(i)}) \quad = -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(\mathbf{x}^{(i)}) \ = 0$$

$$\nabla RSS(w_0, w_1) = \begin{bmatrix} -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)}) \\ -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(\mathbf{x}^{(i)}) \end{bmatrix}$$

Background on the gradient: https://betterexplained.com/articles/vector-calculus-understanding-the-gradient/

Method 1: Closed Form solution

$$\text{RSS}(w_0, w_1) = \sum_{i=1}^{N} \left(y^{(i)} + (-w_0 + -w_1\mathbf{x}^{(i)})\right)^2$$

$$\frac{\partial RSS}{\partial w_0} = \sum_{i=1}^{N} 2(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(-1) \quad = -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)}) = 0$$

$$\frac{\partial RSS}{dw_1} = \sum_{i=1}^{N} 2(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(-\mathbf{x}^{(i)}) \quad = -2\sum_{i=1}^{N}(y^{(i)} - w_0 - w_1\mathbf{x}^{(i)})(\mathbf{x}^{(i)}) \; = 0$$

Remember:

$$\frac{\partial(-a+b)^2}{\partial a} = 2(-a+b)^{2-1}(-1)$$

Simplify

$$\sum_i y^{(i)} = Nw_0 + w_1 \sum_i \mathbf{x}^{(i)}$$

$$\sum_i y^{(i)}\mathbf{x}^{(i)} = w_0 \sum_i \mathbf{x}^{(i)} + w_1 \sum_i \mathbf{x}^{(i)2}$$

Solving these we get the *least squares coefficient estimates* (posted slides will show the calculations)

$$w_0 \; = \bar{y} - w_1\bar{x}$$

$$w_1 \; = \frac{\sum_{i=1}^{N}(\mathbf{x}^{(i)} - \bar{x})(y^{(i)} - \bar{y})}{\sum_{i=1}^{N}(\mathbf{x}^{(i)} - \bar{x})^2}$$

$$\bar{y} = \frac{1}{N}\sum_i y^{(i)} \qquad \bar{\mathbf{x}} = \frac{1}{N}\sum_i \mathbf{x}^{(i)}$$

The next slide was not covered in class

Closed form solution

average sample mpg

average sample horsepower

☑ Take the partial derivatives

$$RSS(\mathbf{w}) = (\epsilon^{(1)})^2 + (\epsilon^{(2)})^2 + \cdots + (\epsilon^{(N)})^2 = \sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 \mathbf{x}^{(i)})^2$$

$$\frac{\partial RSS}{\partial w_0} = -2\sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 \mathbf{x}^{(i)}) \qquad \frac{\partial RSS}{\partial w_1} = -2\sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 x^{(i)})(x^{(i)})$$

☑ Set the partial derivatives to zero and solve for the parameter

I am going to remove this

$$\frac{\partial RSS}{\partial w_0} = \cancel{-2}\sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 \mathbf{x}^{(i)}) = 0 \qquad N w_0 = \sum_{i=1}^{N} y^{(i)} - \sum_{i=1}^{N} w_1 \mathbf{x}^{(i)} \qquad w_0 = \frac{1}{N}\sum_{i=1}^{N} y^{(i)} - \frac{1}{N} w_1 \sum_{i=1}^{N} \mathbf{x}^{(i)} = \underline{\overline{y} - w_1 \overline{x}}$$

$$\frac{\partial RSS}{\partial w_1} = \cancel{-2}\sum_{i=1}^{N} (y^{(i)} - w_0 - w_1 \mathbf{x}^{(i)})(\mathbf{x}^{(i)}) = 0 = \sum_{i=1}^{N} (y^{(i)} - \overline{y} + w_1 \overline{x} - w_1 \mathbf{x}^{(i)})(\mathbf{x}^{(i)}) = \sum_{i=1}^{N} (\mathbf{x}^{(i)} y^{(i)} - \mathbf{x}^{(i)} \overline{y}) - w_1 \sum_{i=1}^{N} ((\mathbf{x}^{(i)})^2 - \mathbf{x}^{(i)} \overline{x})$$

$$w_1 = \frac{\sum_{i=1}^{N} (\mathbf{x}^{(i)} y^{(i)} - \mathbf{x}^{(i)} \overline{y})}{\sum_{i=1}^{N} ((\mathbf{x}^{(i)})^2 - \mathbf{x}^{(i)} \overline{x})}$$

❑Observe that:

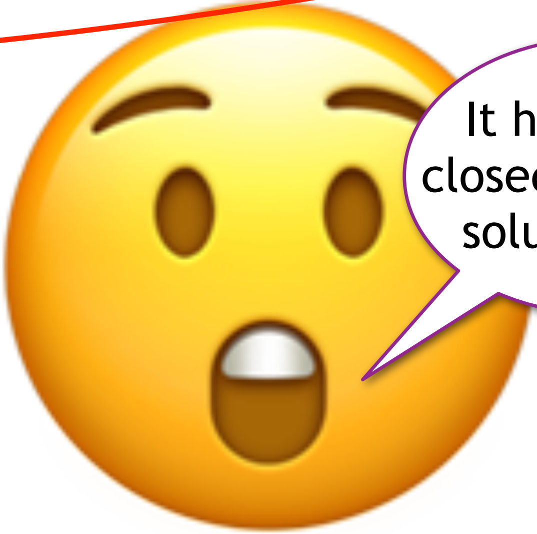
$$\sum_{i=1}^{N} (\overline{x}^2 - \mathbf{x}^{(i)} \overline{x}) = 0 \qquad \sum_{i=1}^{N} (\overline{x}\,\overline{y} - y^{(i)} \overline{x}) = 0$$

$$\left(\sum_{i=1}^{N} \bar{x}^2\right) - \left(\bar{x} \sum_{i=1}^{N} x^{(i)}\right)$$

$$\left(\sum_{i=1}^{N} \bar{x}\,\bar{y}\right) - \left(\bar{x} \sum_{i=1}^{N} y^{(i)}\right)$$

$$w_1 = \frac{\sum_{i=1}^{N} (\mathbf{x}^{(i)} y^{(i)} - \mathbf{x}^{(i)} \overline{y}) + \sum_{i=1}^{N} (\overline{x}\,\overline{y} - y^{(i)} \overline{x})}{\sum_{i=1}^{N} ((\mathbf{x}^{(i)})^2 - \mathbf{x}^{(i)} \overline{x}) + \sum_{i=1}^{N} (\overline{x}^2 - \mathbf{x}^{(i)} \overline{x})} = \frac{\sum_{i=1}^{N} (\mathbf{x}^{(i)} - \overline{x})(y^{(i)} - \overline{y})}{\sum_{i=1}^{N} (\mathbf{x}^{(i)} - \overline{x})^2} = \frac{COV(X, y)}{VAR(X)} = \frac{S_{xy}}{S_{xx}}$$

this is the covariance



It has a closed form solution!

Note that $\sum_{i=1}^{N} \left(\mathbf{x}^{(i)}\right)^2 \neq \bar{x}^2$ while $\sum_{i=1}^{N} \mathbf{x}^{(i)} = N \bar{x}$

Code Notation

- The homework(lab = homework assignment)/Demos will use the following notation
- We will try to be consistent
- Note: Other texts use different notations

Notation

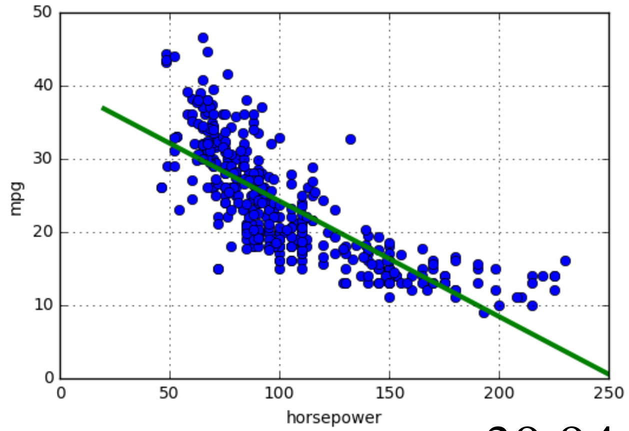
Statistic	Notation	Formula	Python
Mean	$\bar{x}$	$\frac{1}{N}\sum_{i=1}^{N} x^{(i)}$	`xm = np.mean(x)`
(Population) variance	$s_x^2 = s_{xx}$	$\frac{1}{N}\sum_{i=1}^{N} (x^{(i)} - \bar{x})^2$	`sxx = np.mean((x-xm)**2)`
(Population) covariance	s_{xy}	$\frac{1}{N}\sum_{i=1}^{N} (x^{(i)} - \bar{x})(y^{(i)} - \bar{y})$	`sxy= np.mean((x-xm)*(y-ym))`

Auto Example

$$w_0 = \bar{y} - w_1\bar{x}$$

$$w_1 = \frac{\sum_{i=1}^{N}(\mathbf{x}^{(i)}-\bar{x})(y^{(i)}-\bar{y})}{\sum_{i=1}^{N}(\mathbf{x}^{(i)}-\bar{x})^2} = \frac{S_{xy}}{S_{xx}}$$

$$X = \begin{bmatrix} 130.0 \\ 165.0 \\ 150.0 \\ 150.0 \\ 140.0 \\ 198.0 \\ 220.0 \\ 215.0 \\ \vdots \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} 18.0 \\ 15.0 \\ 18.0 \\ 16.0 \\ 17.0 \\ 15.0 \\ 14.0 \\ 14.0 \\ \cdots \end{bmatrix}$$



intercept = $w_0 = 39.94$

slope = $w_1 = -0.16$

```
xm = np.mean(X)
ym = np.mean(y)
```

```
syx = np.mean((y-ym)*(X-xm))
sxx = np.mean((X-xm)**2)
w1 = syx/sxx
w0 = ym - w1*xm
```

$$\bar{x} = (130 + 165 + 150 + \cdots + 215)/392 = 104.47$$

$$\bar{y} = (18 + 15 + 18 + \cdots + 14)/392 = 23.45$$

$$S_{xy} = \big((130-104.47)(18-23.45) + \cdots + (215-104.47)(14-23.45)\big)/392 = -233.26$$

$$S_{xx} = \big((130-104.47)^2 + (165-104.47)^2 + \cdots + (215-104.47)^2)\big)/392 = 1477.79$$

$$w_1 = (-233.26)/1477.79 = -0.16$$

$$w_0 = 23.45 - (-0.16)(104.47) = 39.94$$

```
yhat= w0+w1*X
RSS = np.sum((yhat-y)**2)
MSE = np.mean((yhat-y)**2)# or RSS/N
```

$$RSS(39.94, \; -0.16) = 9385.92$$

$$MSE(39.94, \; -0.16) = 24.00$$

Calculations performed with higher precision

Prediction

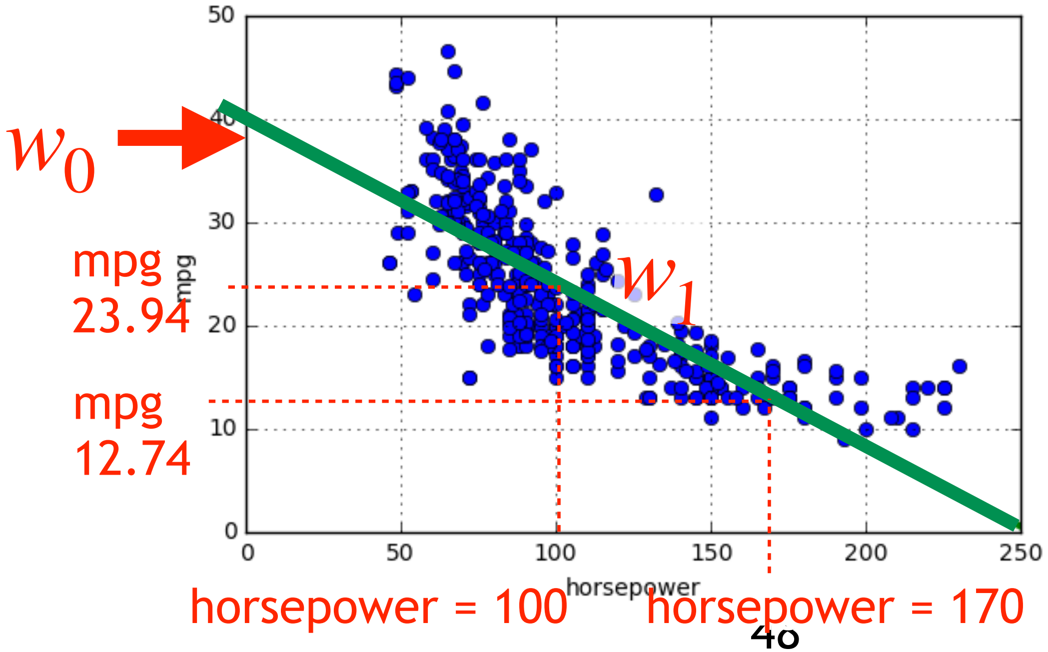
$$\hat{y} = h(\mathbf{x}) = w_0 + w_1\mathbf{x}$$

$w_0 = 39.94$ $\qquad$ $w_1 = -0.16$

MPG ≈ 39.94 + (-0.16) Horsepower

$$y \approx \hat{y} = 39.94 + (-0.16) \cdot 170 = 12.74$$

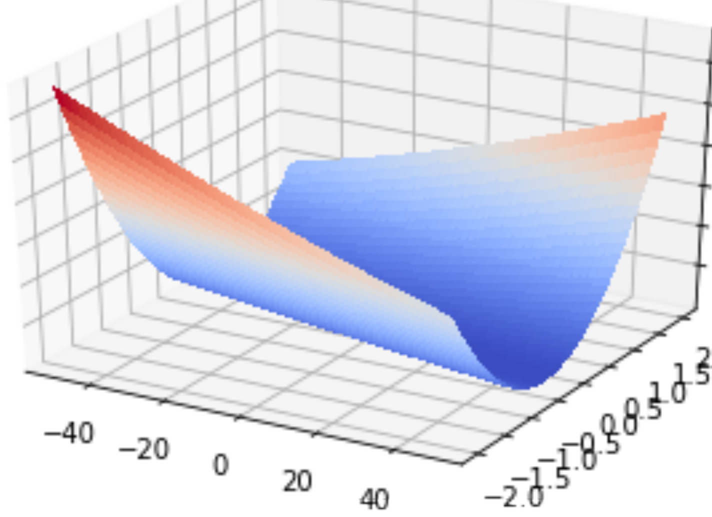
$$y \approx \hat{y} = 39.94 + (-0.16) \cdot 100 = 23.94$$



Outline

- ❑ Motivating Example: Predicting the mpg of a car
- ❑ Model: Linear Model
- ❑ Objective function: Least Squares Fit Problem
- ❑ Global Optimizer: LS Fit Solution
- ❑ Local Optimizer: Gradient Descent
- ❑ Assessing Goodness of Fit
- ❑ Extra Slides: Global Optimizer for multivariate linear regression: Normal Equation

Local Optimization

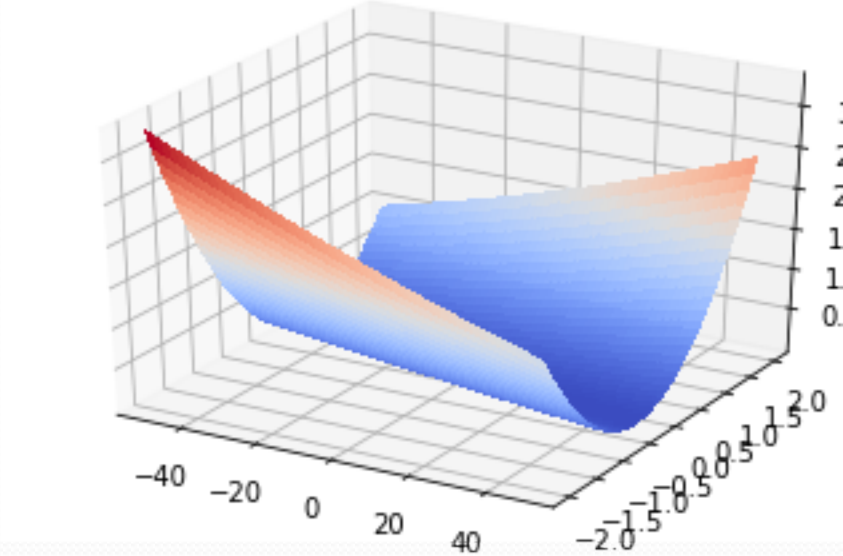


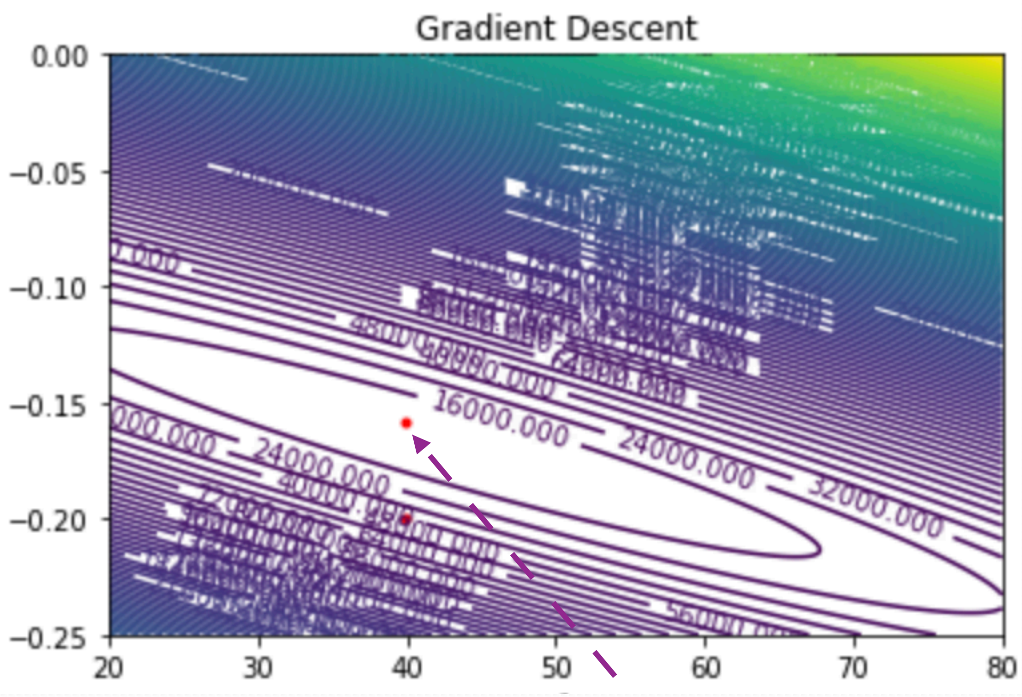
How can we make a small improvement in our parameters?

ANY IDEAS?

Contour Plots

Graphical technique for representing
a 3-D surface in 2-D plane





- **Contour curve**: a 3-D contour curve can be formed on the 3-D surface by finding all the points which have the same Z value (e.g. RSS(w_0, w_1)=k, where k is a constant value)
- **Level curve**: are in the the 2-D plane and can be created by projecting the contour curve onto 2-D plane (e.g.the w_0, w_1 axis)
- **Contour plot**: a 2-D plot containing level curves of a function

$$RSS(39.94, \quad -0.16) = 9385.92$$

Read more at: https://ocw.mit.edu/courses/mathematics/18-02sc-multivariable-calculus-fall-2010/2.-partial-derivatives/part-a-functions-of-two-variables-tangent-approximation-and-optimization/session-25-level-curves-and-contour-plots/MIT18_02SC_notes_15.pdf