CS-UY 4563: Lecture 2
Simple Linear Regression

NYU Tandon School of Engineering, Prof. Christopher Musco

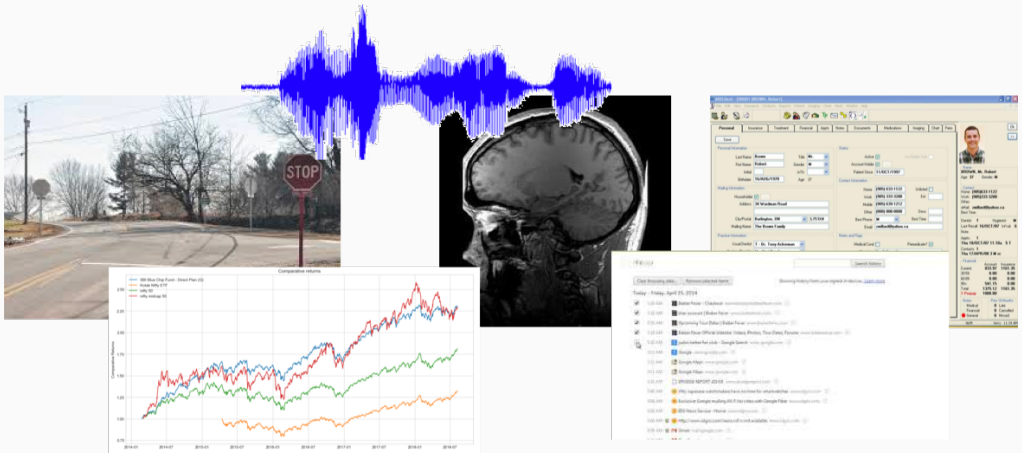
COURSE ADMIN

- Please enroll for Piazza. Only about 60% of class has.
- First lab assignment: `lab_housing_partial.ipynb`
  - Due **next Tuesday, 2/4 at 11:59pm**.
  - Go through the simple regression demonstration `demo_auto_mpg.ipynb`.
  - Turn in entire Jupyter Notebook via NYU Classes.
  - At top of notebook list any collaborators you worked with (as many as you like).
  - There will be a corresponding written homework released shortly.

BASIC GOAL

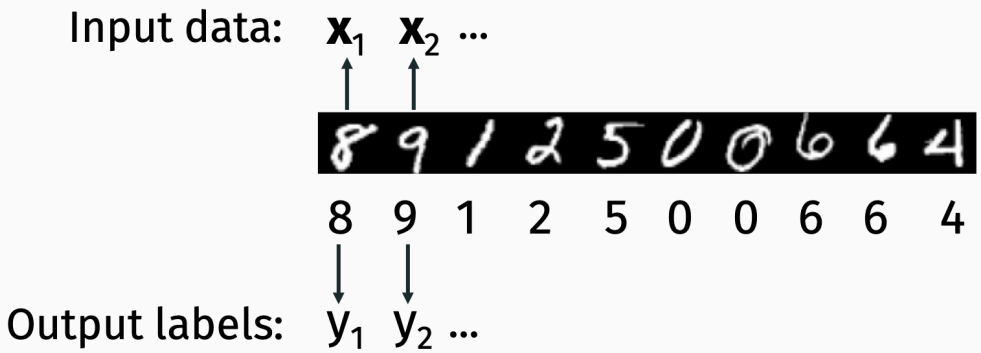
Goal: Develop algorithms to make decisions or predictions based on data.

- **Input:** A single piece of data (an image, audio file, patient healthcare record, MRI scan).



- **Output:** A prediction or decision (this image is a stop sign, this stock will go up 10% next quarter, turn the car right).

Step 1: Collect and label many input/output pairs $(\mathbf{x}_i, y_i)$. For our digit images, we have each $\mathbf{x}_i \in \mathbb{R}^{28\times 28}$ and $y_i \in \{0, 1, \ldots, 9\}$.



This is called the training dataset.

Step 2: Learn from the examples we have.

- Have the computer automatically find some function $f(\mathbf{x})$ such that $f(\mathbf{x}_i) = y_i$ for most $(\mathbf{x}_i, y_i)$ in our training data set (by searching over many possible functions).

In supervised learning every input $\mathbf{x}_i$ in our training dataset comes with a desired output y_i (typically generated by a human, or some other process).

Types of supervised earning:

- **Classification** – predict a discrete class label.
- **Regression** – predict a continuous value.
  - Dependent variable, response variable, target variable, lots of different names for y_i.

Another example of supervised classification: **Face Detection**.

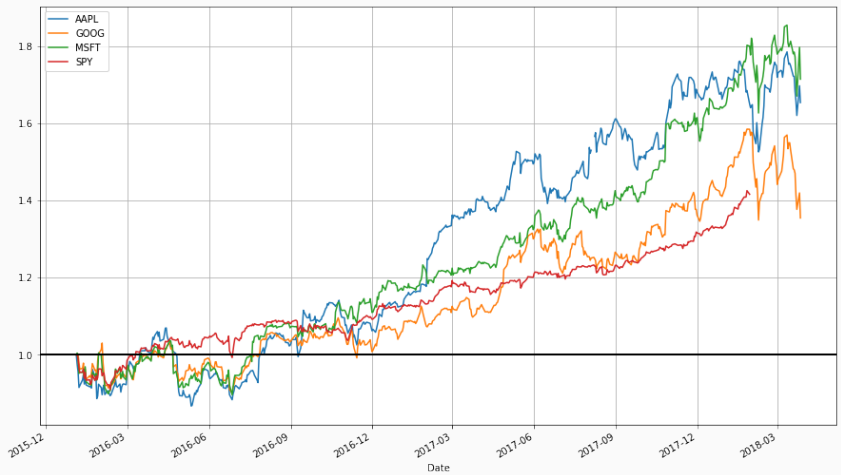
Each input data example $\mathbf{x}_i$ is an image. Each output y_i is 1 if the image contains a face, 0 otherwise.

- Harder than digit recognition, but we now have very reliable methods (used in nearly all digital cameras, phones, etc.)

Other examples of supervised classification:

- Object detection (Input: image, Output: dog or cat)
- Spam detection (Input: email text, Output: spam or not)
- Medical diagnosis (Input: patient data, Output: disease condition or not)
- Credit decision making (Input: financial data, Output: offer loan or not)

Example of supervised regression: **Stock Price Prediction**.



Each input **x** is a vector of metrics about a company (sales volume, PE ratio, earning reports, historical price data).

Each output y_i is the **price of the stock** 3 months in the future.

Other examples of supervised regression:

- Home price prediction (Inputs: square footage, zip code, number of bathrooms, Output: Price)
- Car price prediction (Inputs: make, model, year, miles driven, Output: Price)
- Weather prediction (Inputs: weather data at nearby stations, Output: tomorrows temperature)
- Robotics/Control (Inputs: information about environment and current position at time t, Output: estimate of position at time $t + 1$)

OTHER TYPES OF LEARNING

Later in the class we will talk about other models:

- **Unsupervised learning** (no labels or response variable)
  - Clustering
  - Representation Learning
- **Reinforcement learning**
  - Game playing

You might also hear about semi-supervised learning or active learning – these categories aren't always cut and dry.

SUPERVISED LEARNING

In **supervised learnings** every input $\mathbf{x}_i$ in our training dataset comes with a desired output y_i (typically generated by a human, or some other process).

Types of supervised earning:

- **Classification** – predict a discrete class label.
- **Regression** – predict a continuous value.
  - Dependent variable, response variable, target variable, lots of different names for y_i.

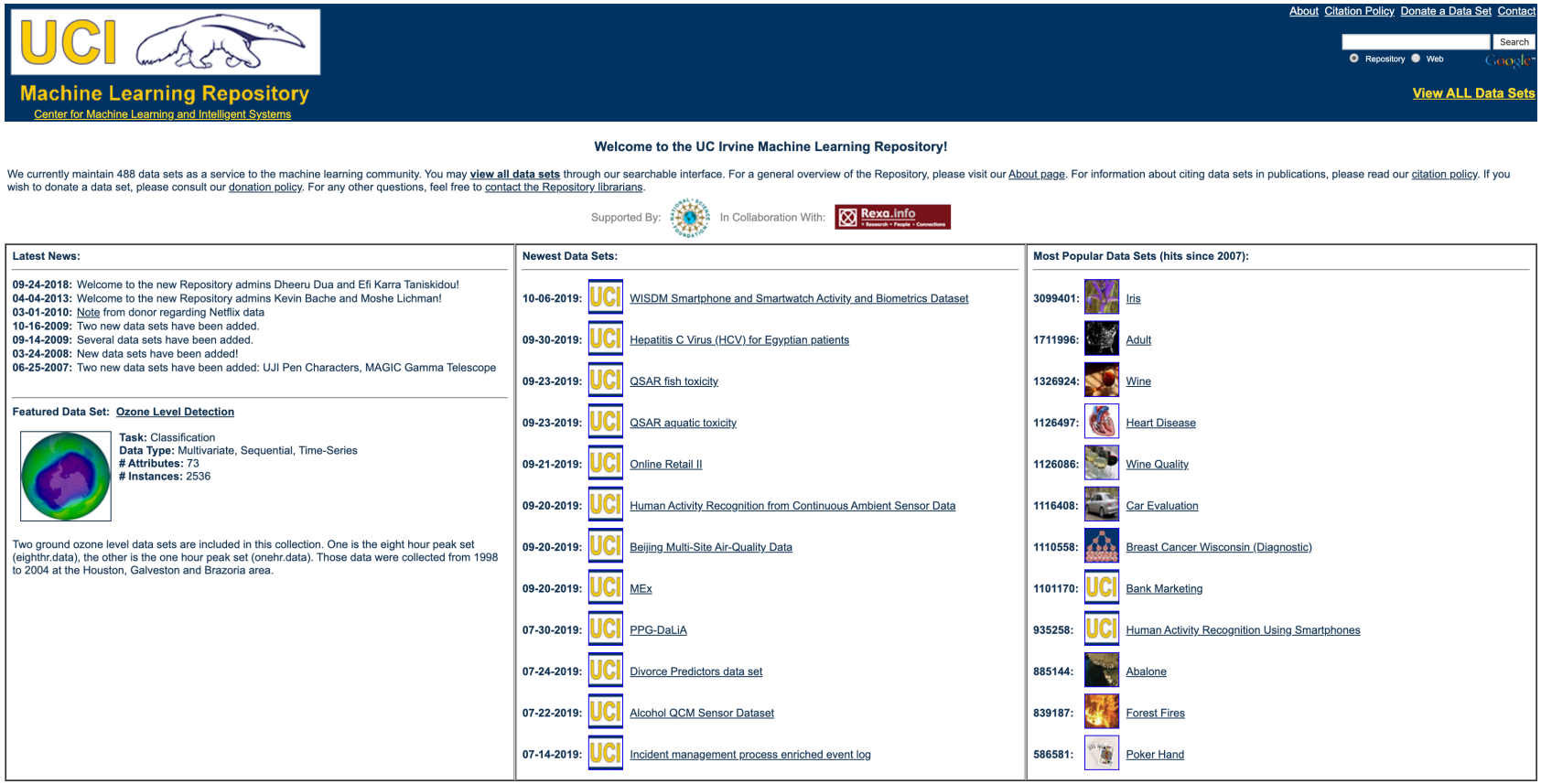
PREDICTING MPG

Motivating example: Predict the highway miles per gallon (MPG) of a car given quantitative information about its engine. Demo in `demo_auto_mpg.ipynb`.

What factors might matter?

PREDICTING MPG

Data set available from the UCI Machine Learning Repository: https://archive.ics.uci.edu/.



UCI Machine Learning Repository
Center for Machine Learning and Intelligent Systems

About Citation Policy Donate a Data Set Contact
Search
Repository Web
View ALL Data Sets

Welcome to the UC Irvine Machine Learning Repository!

We currently maintain 488 data sets as a service to the machine learning community. You may **view all data sets** through our searchable interface. For a general overview of the Repository, please visit our About page. For information about citing data sets in publications, please read our citation policy. If you wish to donate a data set, please consult our donation policy. For any other questions, feel free to contact the Repository librarians.

Supported By: In Collaboration With: Rexa.info

Latest News:

09-24-2018: Welcome to the new Repository admins Dheeru Dua and Efi Karra Taniskidou!
04-04-2013: Welcome to the new Repository admins Kevin Bache and Moshe Lichman!
03-01-2010: Note from donor regarding Netflix data
10-16-2009: Two new data sets have been added.
09-14-2009: Several data sets have been added.
03-24-2008: New data sets have been added!
06-25-2007: Two new data sets have been added: UJI Pen Characters, MAGIC Gamma Telescope

Featured Data Set: Ozone Level Detection

Task: Classification
Data Type: Multivariate, Sequential, Time-Series
 # Attributes: 73
 # Instances: 2536

Two ground ozone level data sets are included in this collection. One is the eight hour peak set (eighthr.data), the other is the one hour peak set (onehr.data). Those data were collected from 1998 to 2004 at the Houston, Galveston and Brazoria area.

Newest Data Sets:

10-06-2019:	WISDM Smartphone and Smartwatch Activity and Biometrics Dataset
09-30-2019:	Hepatitis C Virus (HCV) for Egyptian patients
09-23-2019:	QSAR fish toxicity
09-23-2019:	QSAR aquatic toxicity
09-21-2019:	Online Retail II
09-20-2019:	Human Activity Recognition from Continuous Ambient Sensor Data
09-20-2019:	Beijing Multi-Site Air-Quality Data
09-20-2019:	MEx
07-30-2019:	PPG-DaLiA
07-24-2019:	Divorce Predictors data set
07-22-2019:	Alcohol QCM Sensor Dataset
07-14-2019:	Incident management process enriched event log

Most Popular Data Sets (hits since 2007):

3099401:	Iris
1711996:	Adult
1326924:	Wine
1126497:	Heart Disease
1126086:	Wine Quality
1116408:	Car Evaluation
1110558:	Breast Cancer Wisconsin (Diagnostic)
1101170:	Bank Marketing
935258:	Human Activity Recognition Using Smartphones
885144:	Abalone
839187:	Forest Fires
586581:	Poker Hand

This place is a great resource for projects!

PREDICTING MPG

Datasets from UCI (and many other places) comes as tab, space, or comma delimited files.

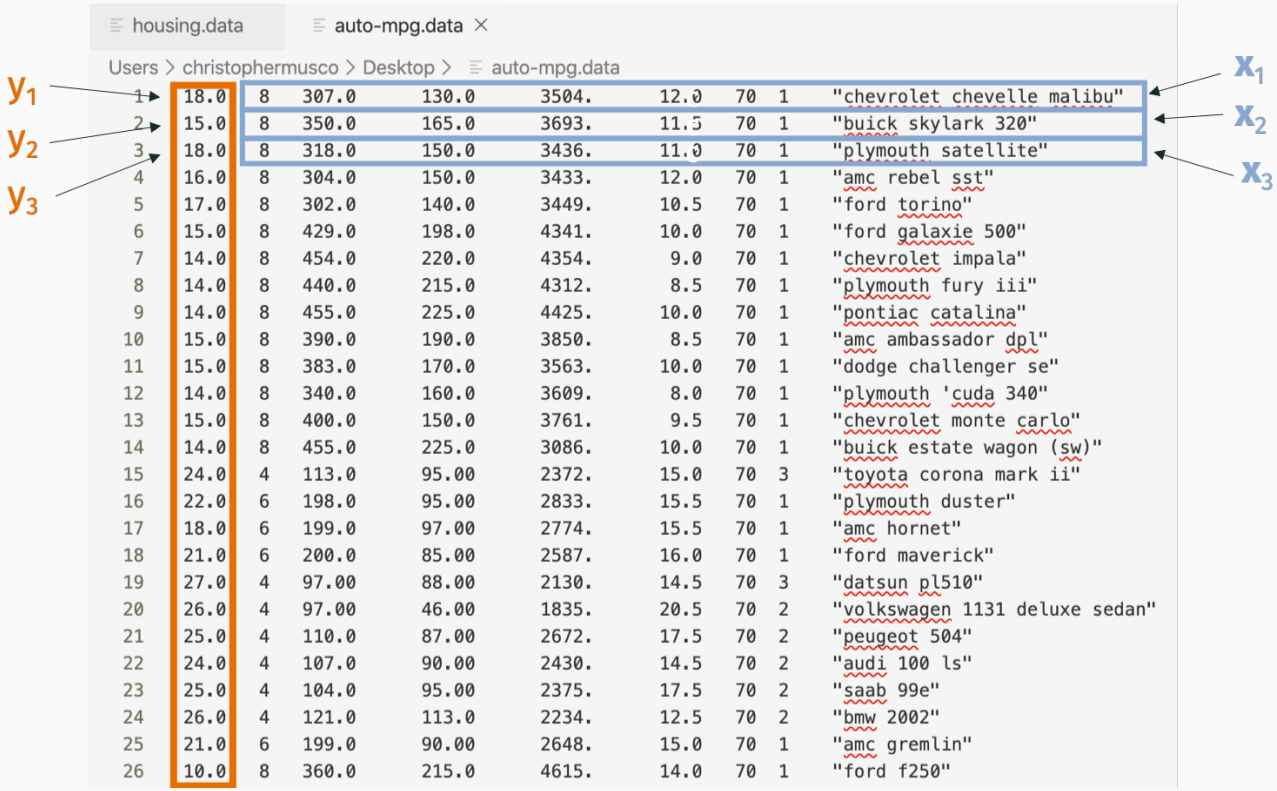
housing.data | auto-mpg.data ×

Users > christophermusco > Desktop > auto-mpg.data

```
  1   18.0   8   307.0      130.0      3504.      12.0   70  1   "chevrolet chevelle malibu"
  2   15.0   8   350.0      165.0      3693.      11.5   70  1   "buick skylark 320"
  3   18.0   8   318.0      150.0      3436.      11.0   70  1   "plymouth satellite"
  4   16.0   8   304.0      150.0      3433.      12.0   70  1   "amc rebel sst"
  5   17.0   8   302.0      140.0      3449.      10.5   70  1   "ford torino"
  6   15.0   8   429.0      198.0      4341.      10.0   70  1   "ford galaxie 500"
  7   14.0   8   454.0      220.0      4354.       9.0   70  1   "chevrolet impala"
  8   14.0   8   440.0      215.0      4312.       8.5   70  1   "plymouth fury iii"
  9   14.0   8   455.0      225.0      4425.      10.0   70  1   "pontiac catalina"
 10   15.0   8   390.0      190.0      3850.       8.5   70  1   "amc ambassador dpl"
 11   15.0   8   383.0      170.0      3563.      10.0   70  1   "dodge challenger se"
 12   14.0   8   340.0      160.0      3609.       8.0   70  1   "plymouth 'cuda 340"
 13   15.0   8   400.0      150.0      3761.       9.5   70  1   "chevrolet monte carlo"
 14   14.0   8   455.0      225.0      3086.      10.0   70  1   "buick estate wagon (sw)"
 15   24.0   4   113.0      95.00      2372.      15.0   70  3   "toyota corona mark ii"
 16   22.0   6   198.0      95.00      2833.      15.5   70  1   "plymouth duster"
 17   18.0   6   199.0      97.00      2774.      15.5   70  1   "amc hornet"
 18   21.0   6   200.0      85.00      2587.      16.0   70  1   "ford maverick"
 19   27.0   4   97.00      88.00      2130.      14.5   70  3   "datsun pl510"
 20   26.0   4   97.00      46.00      1835.      20.5   70  2   "volkswagen 1131 deluxe sedan"
 21   25.0   4   110.0      87.00      2672.      17.5   70  2   "peugeot 504"
 22   24.0   4   107.0      90.00      2430.      14.5   70  2   "audi 100 ls"
 23   25.0   4   104.0      95.00      2375.      17.5   70  2   "saab 99e"
 24   26.0   4   121.0      113.0      2234.      12.5   70  2   "bmw 2002"
 25   21.0   6   199.0      90.00      2648.      15.0   70  1   "amc gremlin"
 26   10.0   8   360.0      215.0      4615.      14.0   70  1   "ford f250"
```

PREDICTING MPG

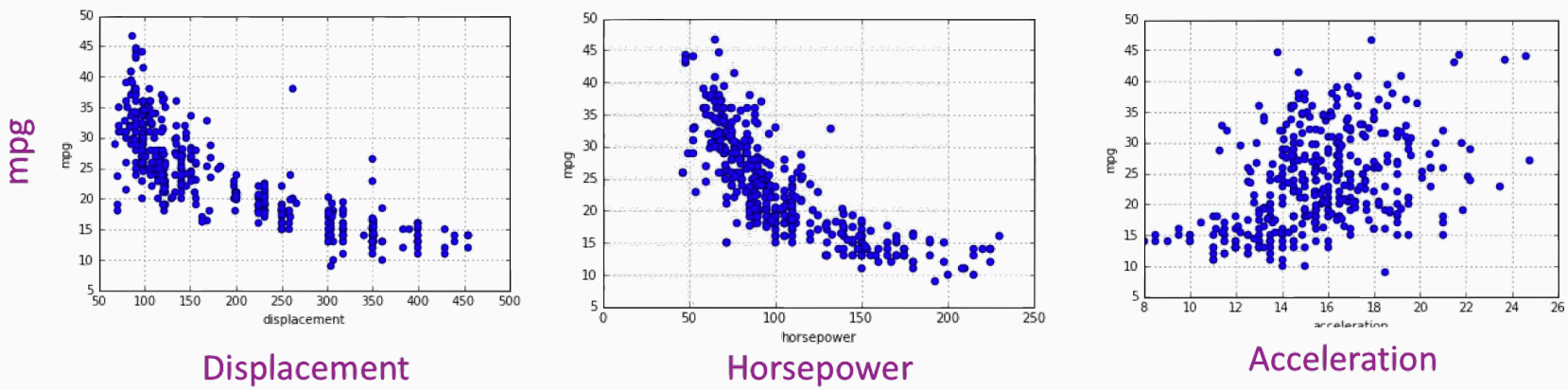
Check dataset description to know what each column means.



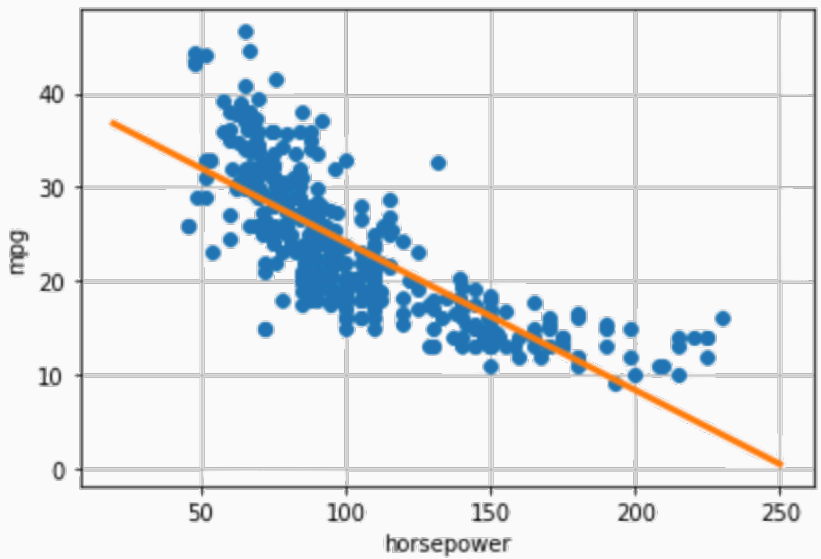
'mpg', 'cylinders','displacement', 'horsepower', 'weight', 'acceleration', 'model year', 'origin', 'car name'

LIBRARIES FOR INITIAL DATA READING

- Use `pandas` for reading data from delimited files. Stores data in a type of table called a “data frame” but this is just a wrapper around a `numpy` array.
- Use `matplotlib` for initial exploration.



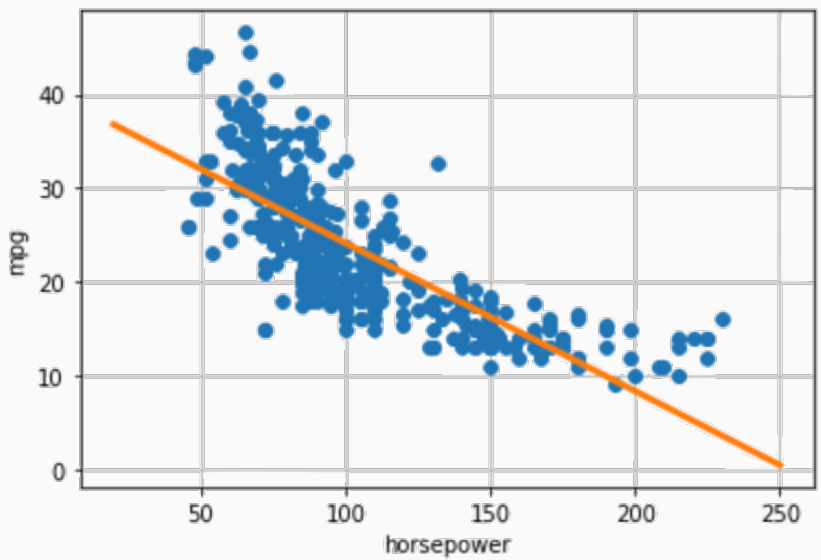
Linear regression from a Machine Learning (not a Statistics) perspective. Our first supervised machine learning model.



Only focus on one predictive variable at a time (e.g. horsepower). This is why it's called simple linear regression.

Dataset:

- $x_1, \ldots, x_n \in \mathbb{R}$ (horsepowers of n cars – this is the predictor/independent variable)
- $y_1, \ldots, y_n \in \mathbb{R}$ (MPG – this is the response/dependent variable)



SUPERVISED LEARNING DEFINITIONS

- **Model $f_{\boldsymbol{\theta}}(x)$:** Class of equations or programs which map input x to predicted output. We want $f_{\boldsymbol{\theta}}(x_i) \approx y_i$ for training inputs.
- **Model Parameters $\boldsymbol{\theta}$:** Vector of numbers. These are numerical nobs which parameterize our class of models.
- **Loss Function $L(\boldsymbol{\theta})$:** Measure of how well a model fits our data. Typically some function of $f_{\boldsymbol{\theta}}(x_1) - y_1, \ldots, f_{\boldsymbol{\theta}}(x_n) - y_n$

Goal: Choose parameters $\boldsymbol{\theta}^*$ which minimize the Loss Function:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta})$$

General Supervised Learning

- Model: $f_{\boldsymbol{\theta}}(x)$
- Model Parameters: $\boldsymbol{\theta}$
- Loss Function: $L(\boldsymbol{\theta})$

Linear Regression

- Model:

$$f_\theta(x_i) = \beta_0 + \beta_1 \cdot x_i$$

- Model Parameters:

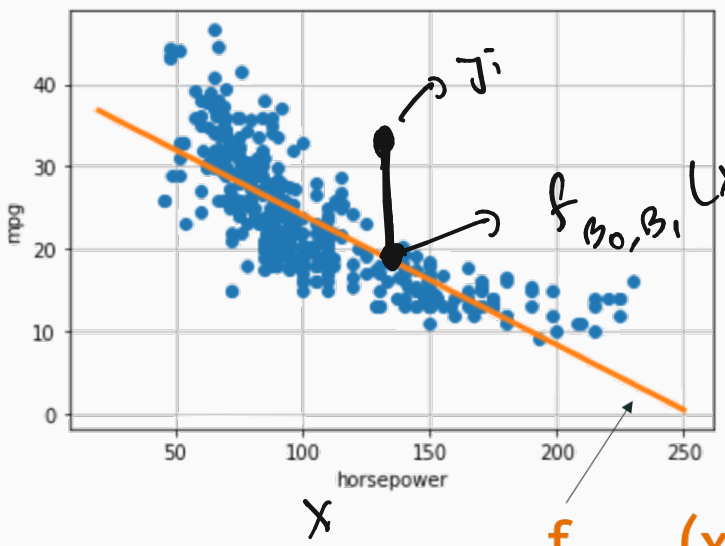
$$\theta = [\beta_0, \beta_1]$$

- Loss Function:

$$L(\theta) = \sum_{i=1}^{n} (y_i - f_\theta(x_i))^2$$

HOW TO MEASURE GOODNESS OF FIT

What is a natural **loss function** for linear regression?



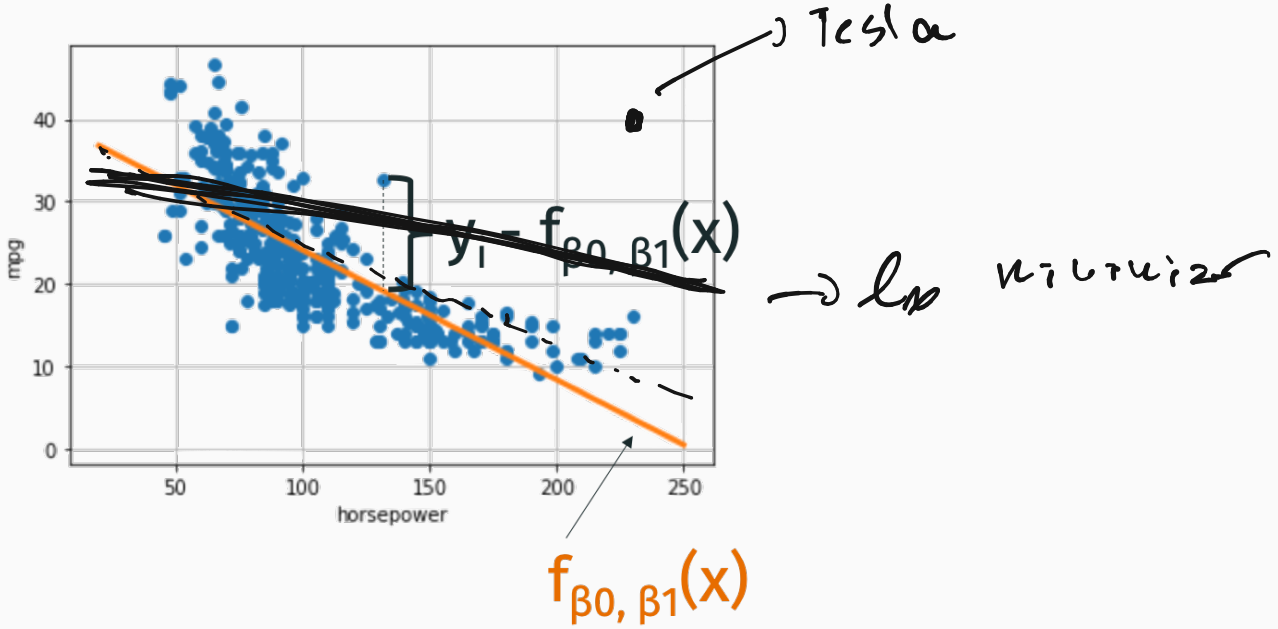
$(y_i - f_{\beta_0,\beta_1}(x_i))^2$

$|y_i - f_{\beta_0,\beta_1}(x_i)|$

$|y_i - f_{\beta_0,\beta_1}(x_i)|^p$

HOW TO MEASURE GOODNESS OF FIT

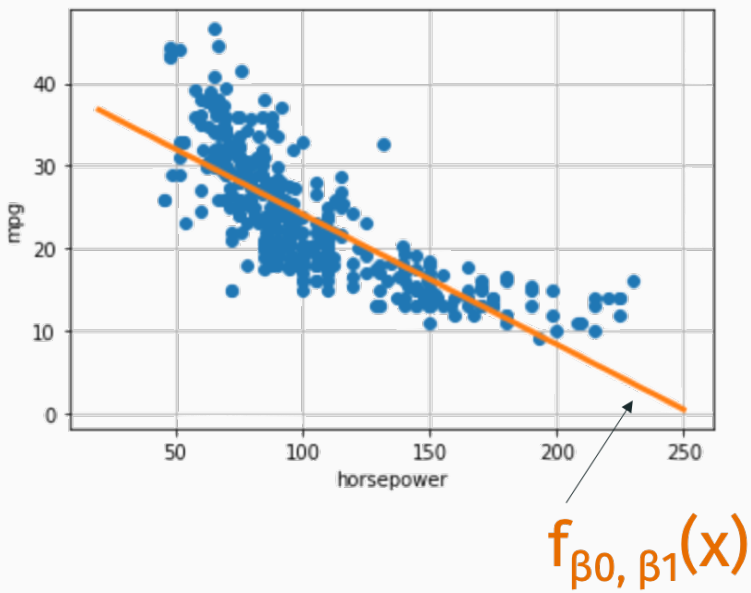
Typical choices are a function of $y_1 - f_{\beta_0,\beta_1}(x_1), \ldots, y_n - f_{\beta_0,\beta_1}(x_n)$



- ℓ_2/**Squared Loss:** $L(\beta_0, \beta_1) = \sum_{i=1}^{n} [y_i - f_{\beta_0,\beta_1}(x_i)]^2$.
- ℓ_1/**Lease absolute deviations:** $L(\beta_0, \beta_1) = \sum_{i=1}^{n} |y_i - f_{\beta_0,\beta_1}(x_i)|$.
- ℓ_∞ **Loss** $L(\beta_0, \beta_1) = \max_{i \in 1,\ldots,n} |y_i - f_{\beta_0,\beta_1}(x_i)|$.

HOW TO MEASURE GOODNESS OF FIT

We're going to start with the Squared Loss/Sum-of-Squares Loss. Also called "Residual Sum-of-Squares (RSS)"



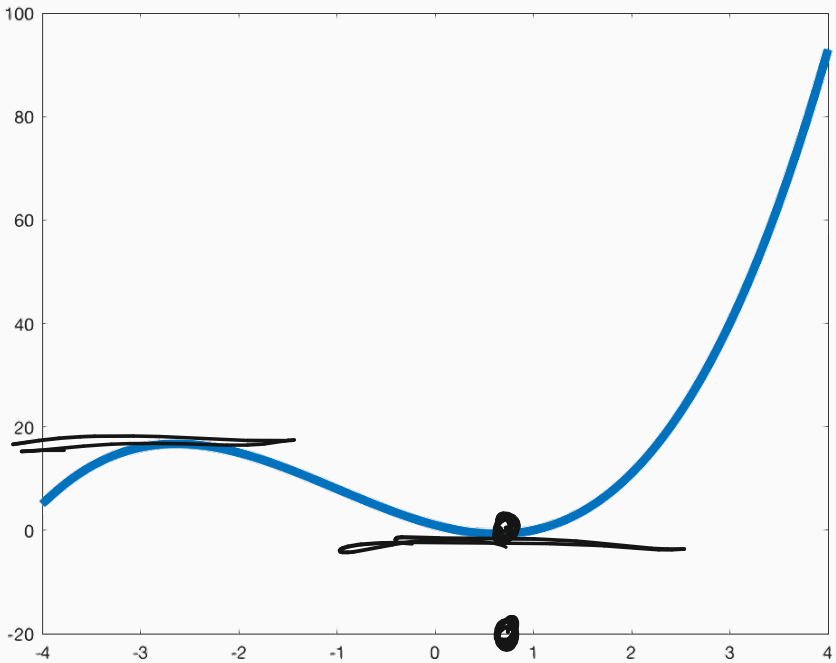
- Relatively robust to outliers.
- Simple to define, leads to simple algorithms for finding β_0, β_1
- Justifications from classical statistics related to assumptions about Gaussian noise. Will discuss later in the course.

General Supervised Learning	Linear Regression
• Model: $f_{\boldsymbol{\theta}}(x)$	• Model: $f_{\beta_0,\beta_1}(x) = \beta_0 + \beta_1 \cdot x$
• Model Parameters: $\boldsymbol{\theta}$	• Model Parameters: β_0, β_1
• Loss Function: $L(\boldsymbol{\theta})$	• Loss Function: $L(\beta_0, \beta_1) = \sum_{i=1}^{n}(y_i - f_{\beta_0,\beta_1}(x_i))^2$

Goal: Choose β_0, β_1 to minimize $L(\beta_0, \beta_1) = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$.

This is the entire job of any Supervised Learning Algorithm.

Univariate function:



$$x^3 + 3 \cdot x^2 - 5 \cdot x + 1$$

- Find all places where derivative $g'(x) = 0$ and check which has the smallest value.

Multivariate function: $L(\beta_0, \beta_1)$

- Find values of β_0, β_1 where all partial derivatives equal 0.
- $\frac{\partial L}{\partial \beta_0} = 0$ and $\frac{\partial L}{\partial \beta_1} = 0$.

MINIMIZING SQUARED LOSS FOR REGRESSION

Multivariate function: $L(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$

- Find values of β_0, β_1 where all partial derivatives equal 0.
- $\frac{\partial L}{\partial \beta_0} = 0$ and $\frac{\partial L}{\partial \beta_1} = 0$.

Some definitions:

- Let $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$. — $\bar{y}$ is the mean of y.
- Let $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$. — $\bar{y}$ is the mean of x.
- Let $\sigma_y^2 = \frac{1}{n} \sum_{i=1}^{n} (y_i - \bar{y})^2$. — σ_y^2 is the variance of y.
- Let $\sigma_x^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$. — σ_x^2 is the variance of x.
- Let $\sigma_{xy} = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y})$. — σ_{xy} is the covariance.

Claim: $L(\beta_0, \beta_1)$ is minimized when:

- $\beta_1 = \sigma_{xy} / \sigma_x^2$
- $\beta_0 = \bar{y} - \beta_1 \bar{x}$

PROOF

$$L(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$$

$$0 = \frac{\partial L}{\partial \beta_0} = \sum_{i=1}^{n} \frac{\partial L}{\partial \beta_0} (y_i - \beta_0 - \beta_1 x_i)^2$$ → used linearity of derivative

$$= \sum_{i=1}^{n} 2(y_i - \beta_0 - \beta_1 x_i) \cdot (-1)$$ → used chain rule

$$0 = -\cancel{2} \sum_{i=1}^{n} y_i - \beta_0 - \beta_1 x_i$$

$$= \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} \beta_0 - \sum_{i=1}^{n} \beta_1 x_i$$

$$0 = \sum_{i=1}^{n} y_i - n\beta_0 - \beta_1 \sum_{i=1}^{n} x_i$$

$$\beta_0 = \frac{\sum_{i=1}^{n} y_i - \beta \sum_{i=1}^{n} x_i}{n} = \bar{y} - \beta_1 \bar{x}$$

PROOF

From prior page, $L(\beta_0, \beta_1) = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2$

achieves its minimum when $\beta_0 = \bar{y} - \beta_1 \bar{x}$.

So, to find the optimal β_1, it suffices to minimize

$$\tilde{L}(\beta_1) = L(\bar{y} - \beta_1 \bar{x}, \beta_1) = \sum_{i=1}^{n} (y_i - \bar{y} + \beta_1 \bar{x} - \beta_1 x_i)^2$$

$$= \sum_{i=1}^{n} (y_i - \bar{y} + \beta_1 (\bar{x} - x_i))^2$$

Setting derivative to 0:

$$0 = \frac{d\tilde{L}}{d\beta_1} = \sum_{i=1}^{n} 2 (y_i - \bar{y} + \beta_1 (\bar{x} - x_i))(\bar{x} - x_i)$$

→ chain rule used again

$$0 = \sum_{i=1}^{n} (y_i - \bar{y})(\bar{x} - x_i) + \beta_1 \sum_{i=1}^{n} (\bar{x} - x_i)^2$$

$$0 = -n\, \mathrm{Cov}_{xy} + \beta_1 n\, \mathrm{Var}_x$$

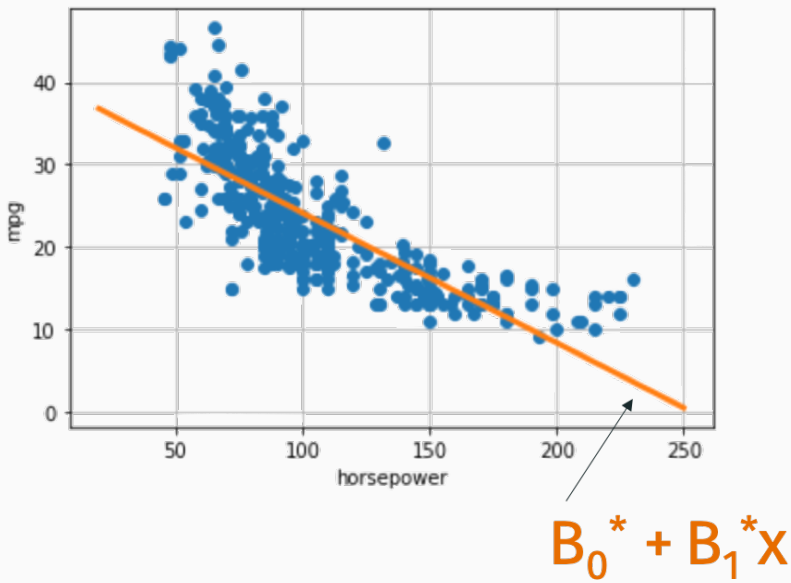
Solving for β_1:

$$\beta_1 = \mathrm{Cov}_{xy} / \mathrm{Var}_x .$$

MINIMIZING SQUARED LOSS FOR REGRESSION

Takeaways:

- Minimizing functions is often easy with calculus.
- Tools we will see again: **linearity of derivatives**, **chain rule**.
- Simple closed form formula for optimal parameters β_0^* and β_1^* for squared-loss!



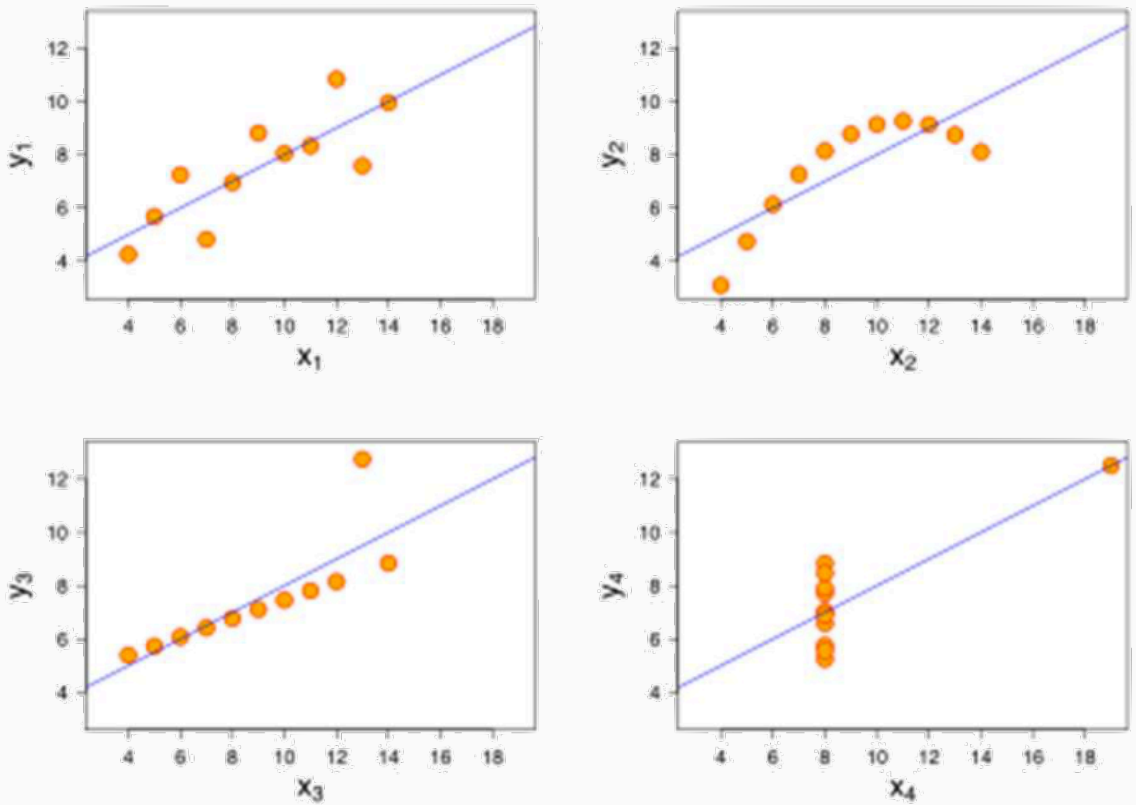
Let $L(\beta_0, \beta_1) = \sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2$.

$$R^2 = 1 - \frac{L(\beta_0, \beta_1)}{n\sigma_y^2}$$

is exactly the R^2 value you may remember from statistics.

The smaller the loss, the closer R^2 is to 1, which means we have a better regression fit.

Many reasons you might get a poor regression fit:

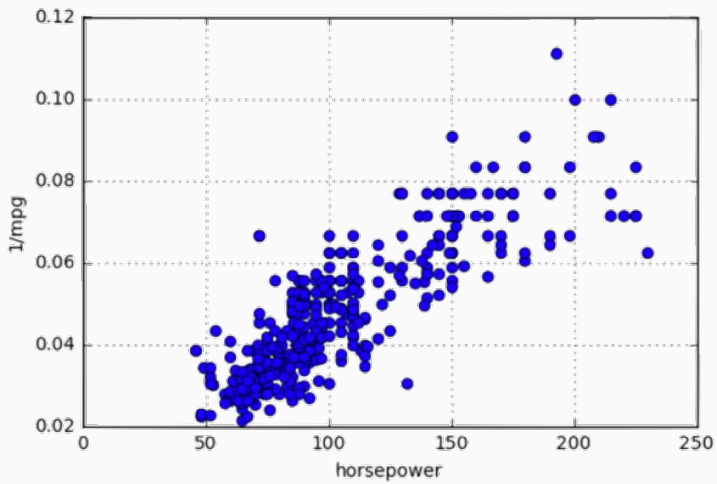


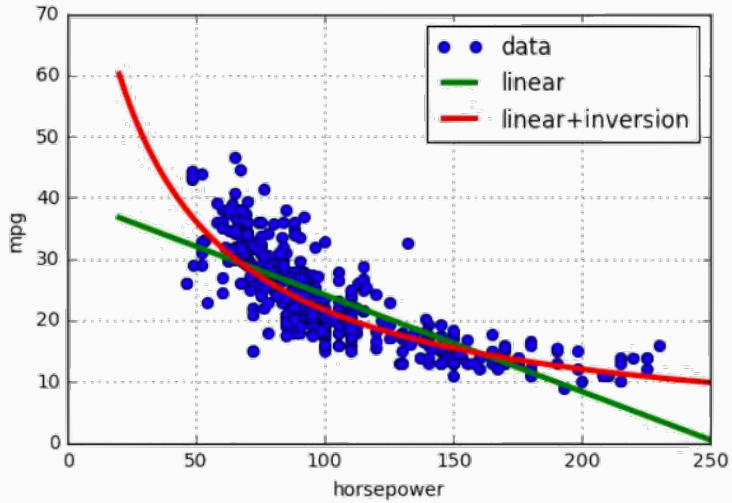
A FEW COMMENTS

Some of these are fixable!

- Remove outliers, use more robust loss function.
- Non-linear model transformation.

Fit the model $\frac{1}{\text{mpg}} \approx \beta_0 + \beta_1 \cdot \text{horsepower}$.





Much better fit, same exact learning algorithm!