Linear Regression

Data Science Decal

Hosted by Machine Learning at Berkeley

ML@B

Agenda

Background

Model Estimation

Assumptions

Model Testing

Next Steps

Background

Where are we?

	Continuous	Discrete
Supervised	**Regression**	Classification
Unsupervised	Dimensionality Reduction	Clustering

- Suppose we have data
  - Want to model relationships and make **predictions**
- The data has **continuous** labels (y)
  - i.e. prices, heights, miles per gallon, etc.
- The data has a set of **explanatory** variables (x_i)
  - i.e. sales, weights, engine power, etc.
- How does a computer make predictions?

Why linear regression?

ML@B

- Regression is one of the most commonly used methods by data scientists
- It is simple, fast, interpretable, and **powerful**
- The techniques we use here are widely applicable
- It is practical!
  - (Physics) Ohm's law, Hooke's law, Charles's law
  - (Economics) Okun's law

Linear regression

ML@B

- Suppose we have p predictor variables x_1, x_2, ..., x_p
- We can approximate y as a linear function of the $x_i's$:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \cdots + \theta_p x_p$$

- θ_i's are the **parameters** (also called **weights**) which we need to estimate
- We introduce $x_0 = 1$ for simplicity so that:

$$y = \sum_{i=1}^{p} \theta^T x$$

Warm-up: predicting house prices

ML@B

- Suppose we have the following data about houses:

Price	**# of Square Feet**	**# of Bedrooms**
221,900	1180	3
538,000	2570	3
⋮	⋮	⋮
1,225,000	5420	4

- Let's predict the price of a house from the number of square feet it has
- Our linear model has the form:

$$h_\theta(sqft) = \theta_0 + \theta_1 sqft$$

Model Estimation

How do we choose θ?

ML@B

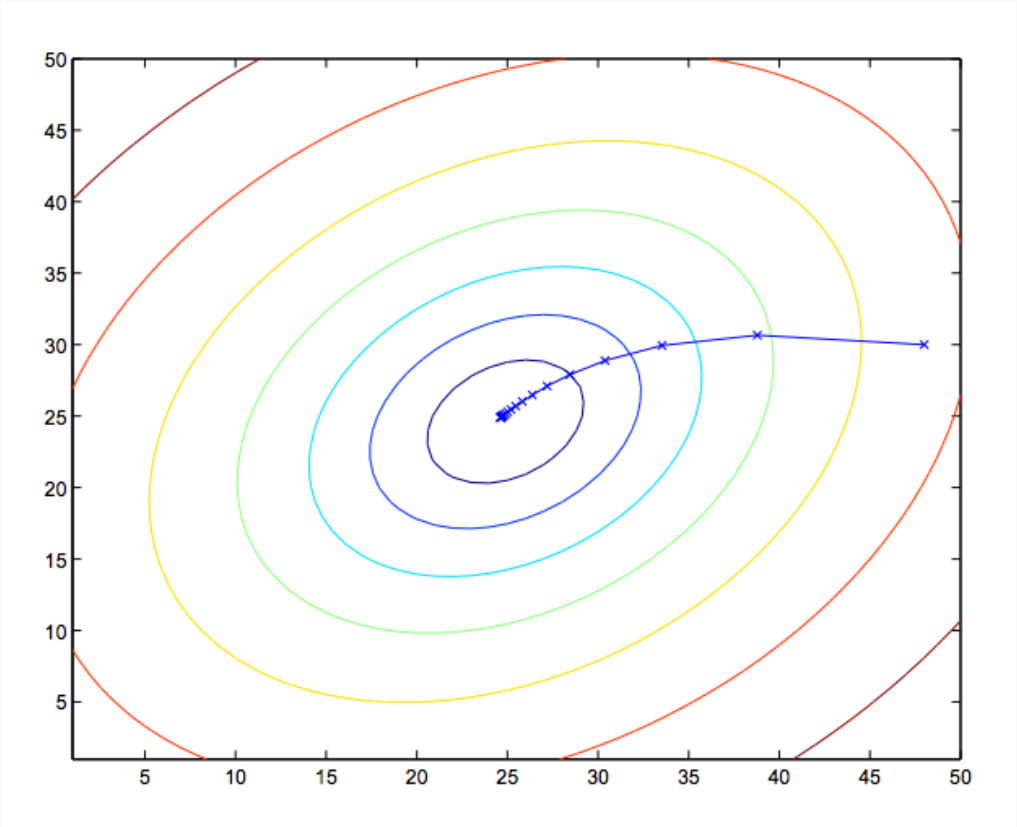
- **Goal**: have $h_\theta(x)$ be as close to y as possible
- We can translate this goal into mathematics by defining the **cost function**

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{n}(h_\theta(x^{(i)}) - y^{(i)})^2$$

- $J(\theta)$ sums the squared **residuals**
- To have an accurate model, we want to **minimize** $J(\theta)$

Minimizing the cost function

- **Idea**: choose θ to minimize $J(\theta)$
- We can use a search algorithm that follows the scheme:
  - Choose an initial guess for θ
  - Repeatedly update θ to make $J(\theta)$ smaller
  - Keep doing this until $J(\theta)$ reaches its minimum



Updating θ to minimize $J(\theta)$

ML@B

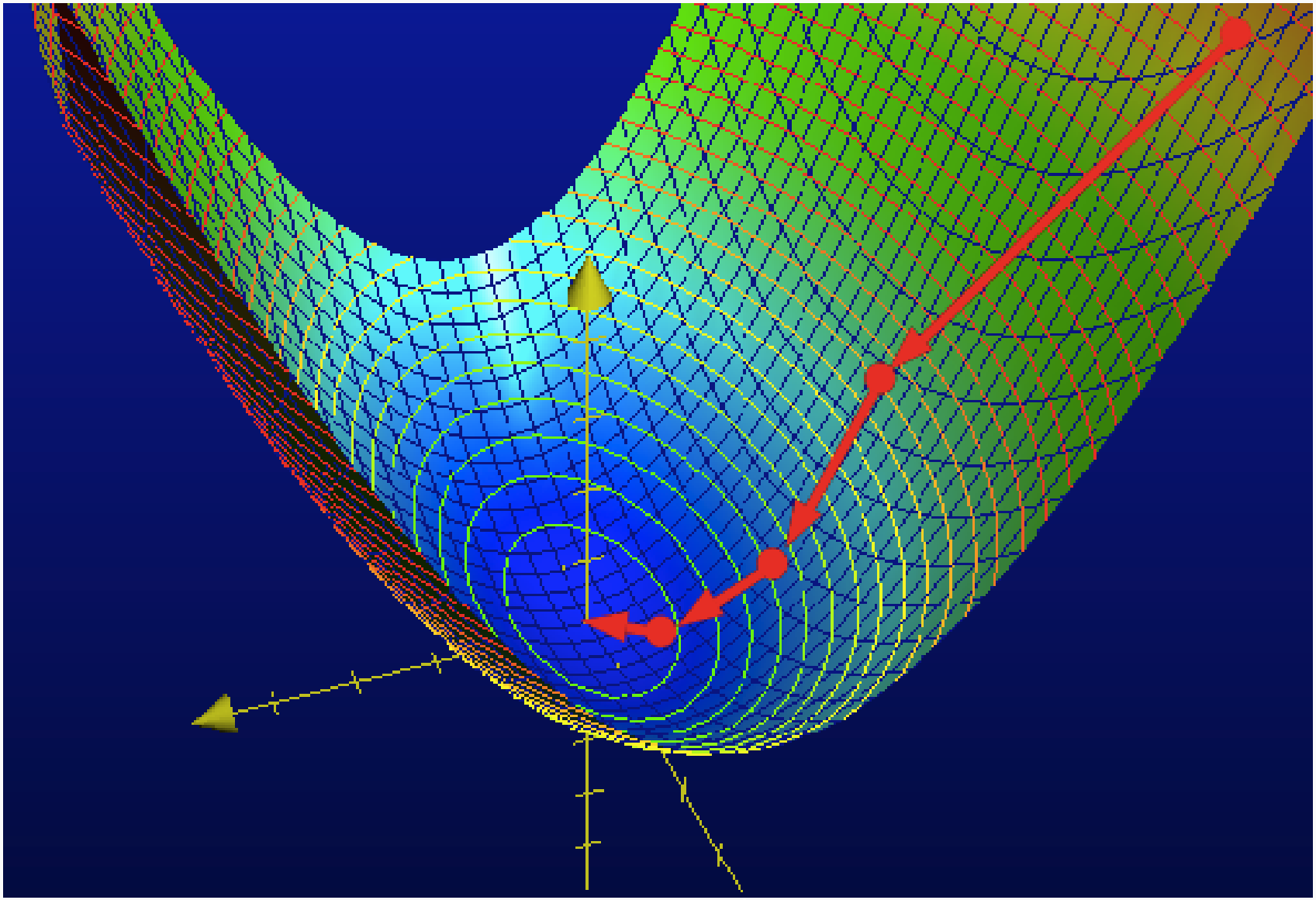
- **Note**: $J(\theta)$ is a convex quadratic function (has nice properties)
- From Math 53: the direction of greatest increase is the same direction of the gradient vector
- **Idea**: let's update θ by traversing the opposite direction instead
- This scheme is known as **gradient descent**

$$\theta \leftarrow \theta - \epsilon \nabla_\theta J(\theta)$$

- ϵ is called the **learning rate**

Visualizing gradient descent

ML@B



Deriving the update rule

- Let's start with the case where we only have one training example $(x^{(1)}, y^{(1)})$

$$\begin{aligned}\nabla_\theta J(\theta) &= \nabla_\theta \frac{1}{2}(h_\theta(x^{(1)}) - y^{(1)})^2 \\ &= 2(\frac{1}{2})(h_\theta(x^{(1)}) - y^{(1)})\nabla_\theta(h_\theta(x^{(1)}) - y^{(1)}) \\ &= (h_\theta(x^{(1)}) - y^{(1)})\nabla_\theta(\theta^T x^{(1)} - y^{(1)}) \\ &= (h_\theta(x^{(1)}) - y^{(1)})x^{(1)}\end{aligned}$$

- For a single training example, the update rule is:

$$\theta \leftarrow \theta - \epsilon(y^{(1)} - h_\theta(x^{(1)}))x^{(1)}$$

Gradient descent for linear regression

- For n training examples:

$$\theta_j \leftarrow \theta_j - \epsilon \sum_i^n (h_\theta(x^{(i)}) - y^{(1)}) x_j^{(i)}$$

  for $j = 1, \ldots, p$ and$i = 1, \ldots, n$

- This rule is also called the **LMS** update rule ("least mean squares")
- Size of update is proportional to the residual term $(y^{(i)} - h_\theta(x^{(i)}))$
- If the prediction $h_\theta(x^{(i)})$ is close the actual $y^{(i)}$ then the parameters θ shouldn't need much changing

Gradient descent as an optimization algorithm

ML@B

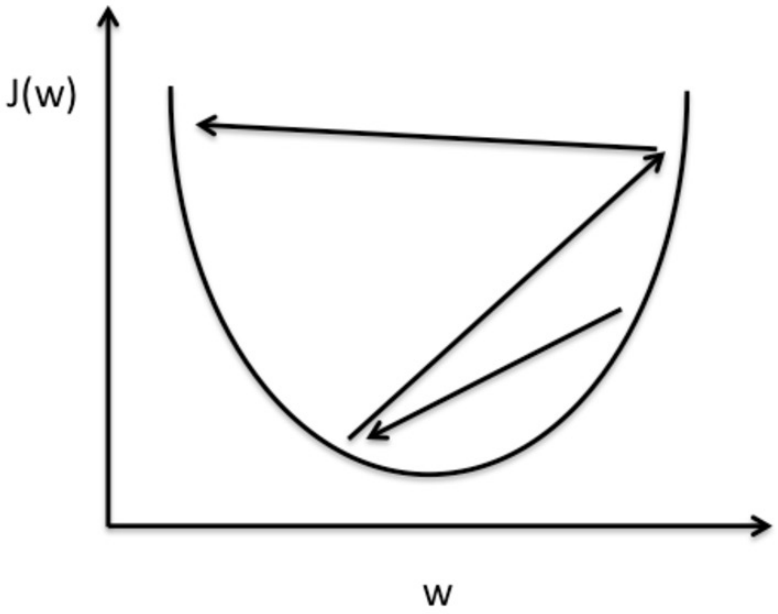
Stochastic gradient descent for linear regression

While $J(\theta)$ is not minimized:

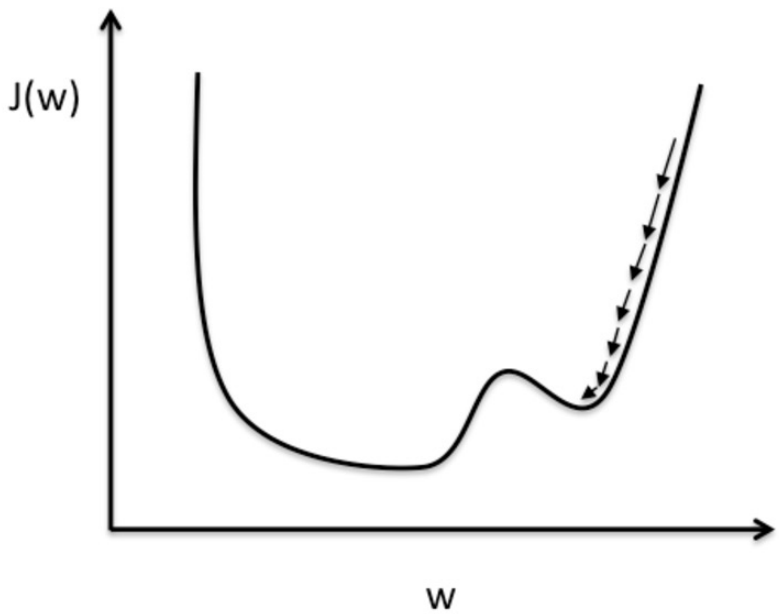
For $i = i, \ldots, n$:

$$\theta_j \leftarrow \theta_j - \epsilon(h_\theta(x^{(i)}) - y^{(i)})x_j^{(i)} \qquad \text{(for each j)}$$

Choosing the learning rate



Large learning rate: Overshooting.



Small learning rate: Many iterations until convergence and trapping in local minima.

Linear regression can be solved in an easier way...

ML@B

- SGD is the basis for many optimization algorithms
- Not necessary for linear regression, as there exists a closed form solution
- We can find the optimal θ by solving the **normal equations**

Another interpretation

- Recall the equation for a linear model:

$$h_\theta(x_i) = \theta_0 + \theta_1 x_{i1} + \cdots + \theta_p x_{ip}$$

- The outcome, y, which we observe can be thought of as:

$$y_i = h_\theta(x_i) + \epsilon_i$$

  where ϵ is some unobserved error
- We don't know the true θ is, so we estimate it with $\hat{\theta}$
- Our predictions for test points are then

$$\hat{y} = h_{\hat{\theta}}(x)$$

Linear regression in matrix form

- We can rewrite linear regression as

$$\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \begin{pmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{pmatrix} \begin{pmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_p \end{pmatrix} + \begin{pmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{pmatrix}$$

- In compressed notation:

$$\vec{y} = X\vec{\theta} + \vec{\epsilon}$$

- Here, we're using capital letters to represent matrices, and arrows to represent vectors

Estimating θ

- We want our estimate, $\hat{\theta}$ to be accurate
- We can be accurate by trying to minimize error
- We can be accurate by minimizing our residuals

$$e_i = y_i - \vec{\theta}^T x_i$$

- More mathematically convenient to minimize squared residuals
- That is,

$$\hat{\vec{\theta}} = argmin_{\vec{\theta}} \sum_{i=1}^{n} (y_i - (\theta_0 + \theta_1 x_{i1} + \cdots + \theta_p x_{ip}))^2$$

Estimation (least squares)

$$\hat{\theta} = argmin_{\vec{\theta}} ||\vec{y} - X\vec{\theta}||_2^2$$

$$= argmin_{\vec{\theta}} (\vec{y} - X\vec{\theta})^T (\vec{y} - X\vec{\theta})$$

$$= argmin_{\vec{\theta}} \vec{y}^T \vec{y} - 2\vec{y}X\vec{\theta} + \vec{\theta}^T X^T X\vec{\theta}$$

Let $Q = ||\vec{y} - X\vec{\theta}||_2^2$

Taking the derivative with respect to the vector $\vec{\theta}$:

$$\frac{\partial Q}{\partial \vec{\theta}} = 2X^T X\vec{\theta} - 2X^T \vec{y} = 0$$

$$\boxed{\hat{\theta} = (X^T X)^{-1} X^T \vec{y}}$$

Some facts

- $\hat{\theta}$ is indeed a minimizer (the second derivative is negative)
- Gauss Markov Theorem: $\hat{\theta}$ is BLUE (best linear unbiased estimator)
- The residuals are:

$$\vec{e} = (\vec{y} - \hat{\vec{y}}) = (I_{n\times x} - X(X^T X)^{-1} X^T)\vec{y}$$

$$[\hat{\vec{y}} = X\hat{\theta} = X(X^T X)^{-1} X^T Y]$$

- $\hat{\theta}$ is a random variable and thus has variance:

$$Var(\hat{\theta}) = Var((X^T X)^{-1} X^T Y) = (X^T X)^{-1} X^T Var(\vec{y}) X (X^T X)^{-1}$$

$$= \sigma^2 (X^T X)^{-1}$$

$$[Var(\vec{y}) = \sigma^2 I_{n\times n}]$$

Model interpretation

ML@B

- Recall, once we have our estimate $\hat{\theta}$, we can predict new x's using:

$$\hat{y}_i = \hat{\theta}_0 + \hat{\theta}_1 x_{i1} + \cdots + \hat{\theta}_p x_{ip}$$

- In matrix notation:

$$\hat{\vec{y}} = X\hat{\vec{\theta}}$$

- For a one unit increase in x_{ik}, we expect y_i to, **on average** increase by $\hat{\theta}_k$
- If we take the log of the independent variables, the dependent variable, or both, then the above interpretation changes to involve **percent changes**

Assumptions

Assumptions

ML@B

- Regression is a good summary of data, assuming the data has some key properties
- We need to know what those assumptions are, how to test for them, and what to do when they fall apart

Assumptions: what are they?

- Linearity
- Normality of errors

$$\epsilon_i \sim N(0, \sigma^2)$$

- Homoscedasticity (constant variance)

$$Var(\epsilon_i) = \sigma^2 \neq \sigma^2(x)$$

- Independence of errors

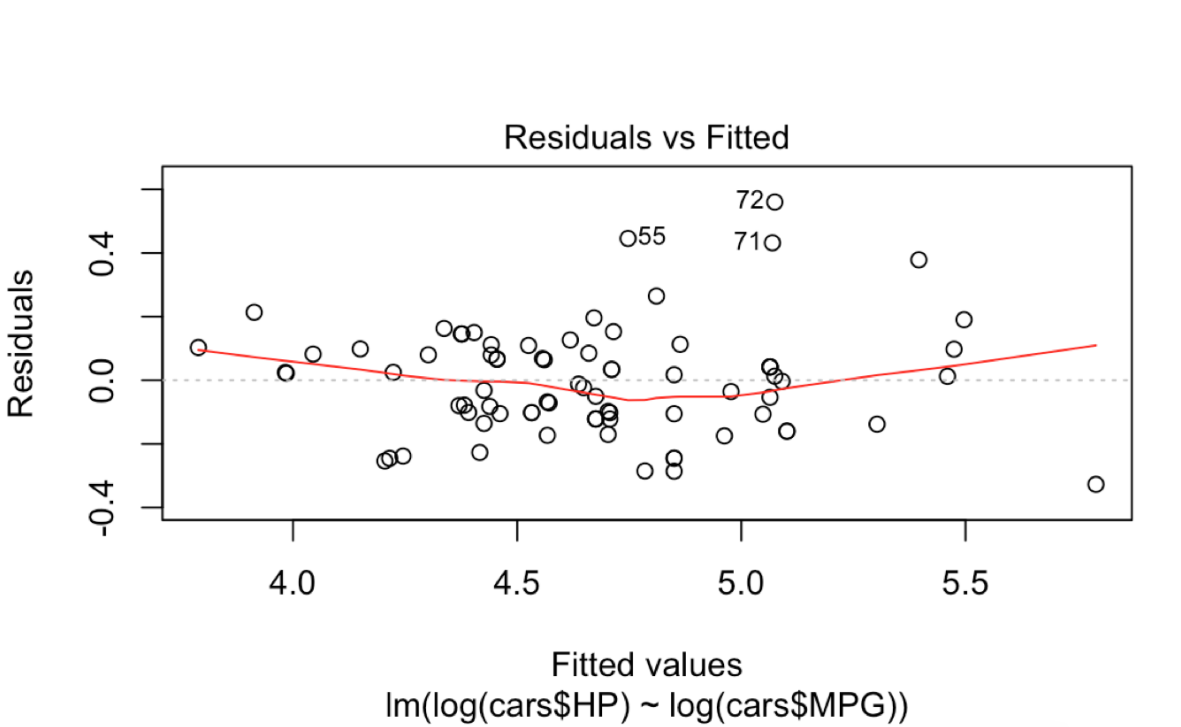
$$\epsilon_i \perp\!\!\!\perp \epsilon_j \qquad \forall i \neq j$$

Assumptions: how do we check them?

ML@B

Linearity

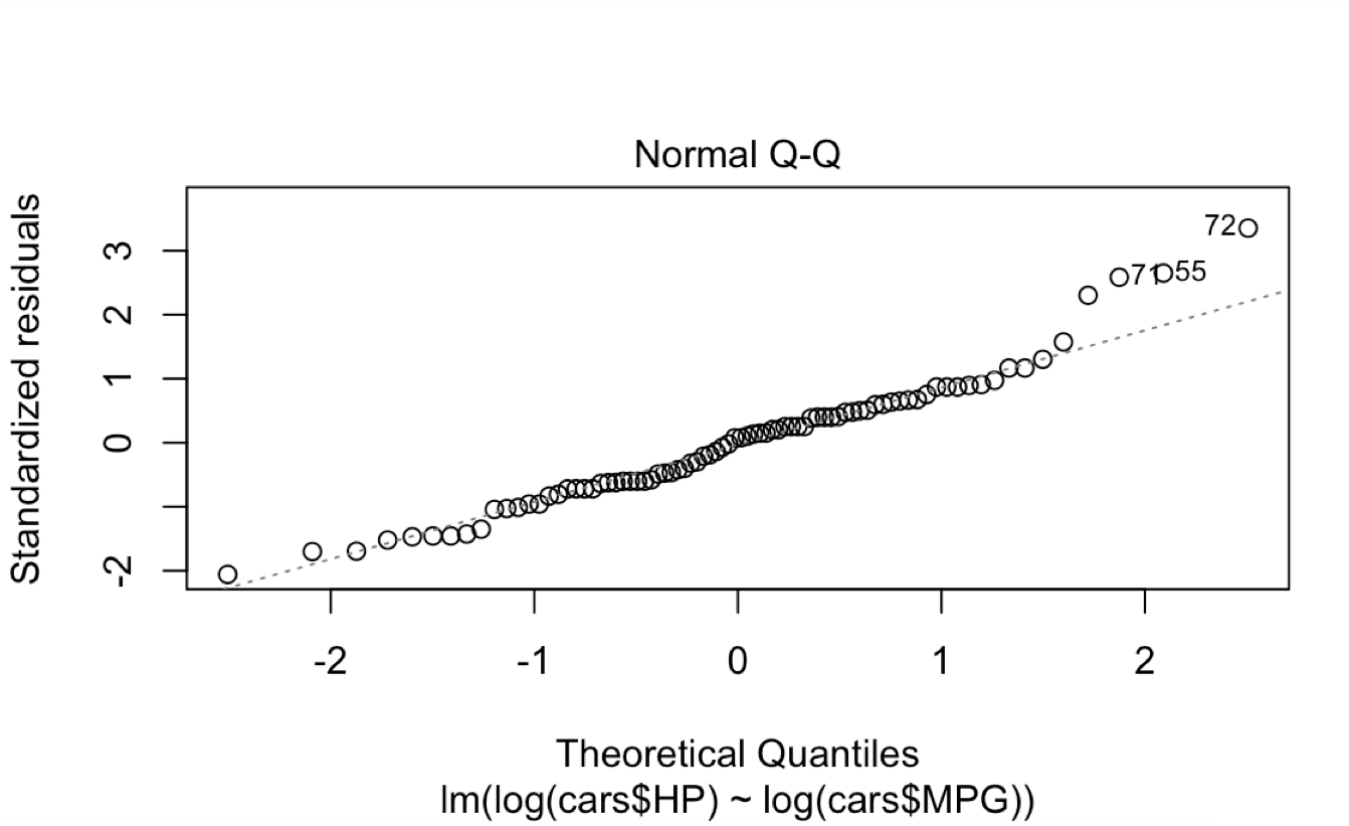
- Scatter plot of Y vs. standardized residuals should have no pattern



Assumptions: how do we check them?

Normality of errors

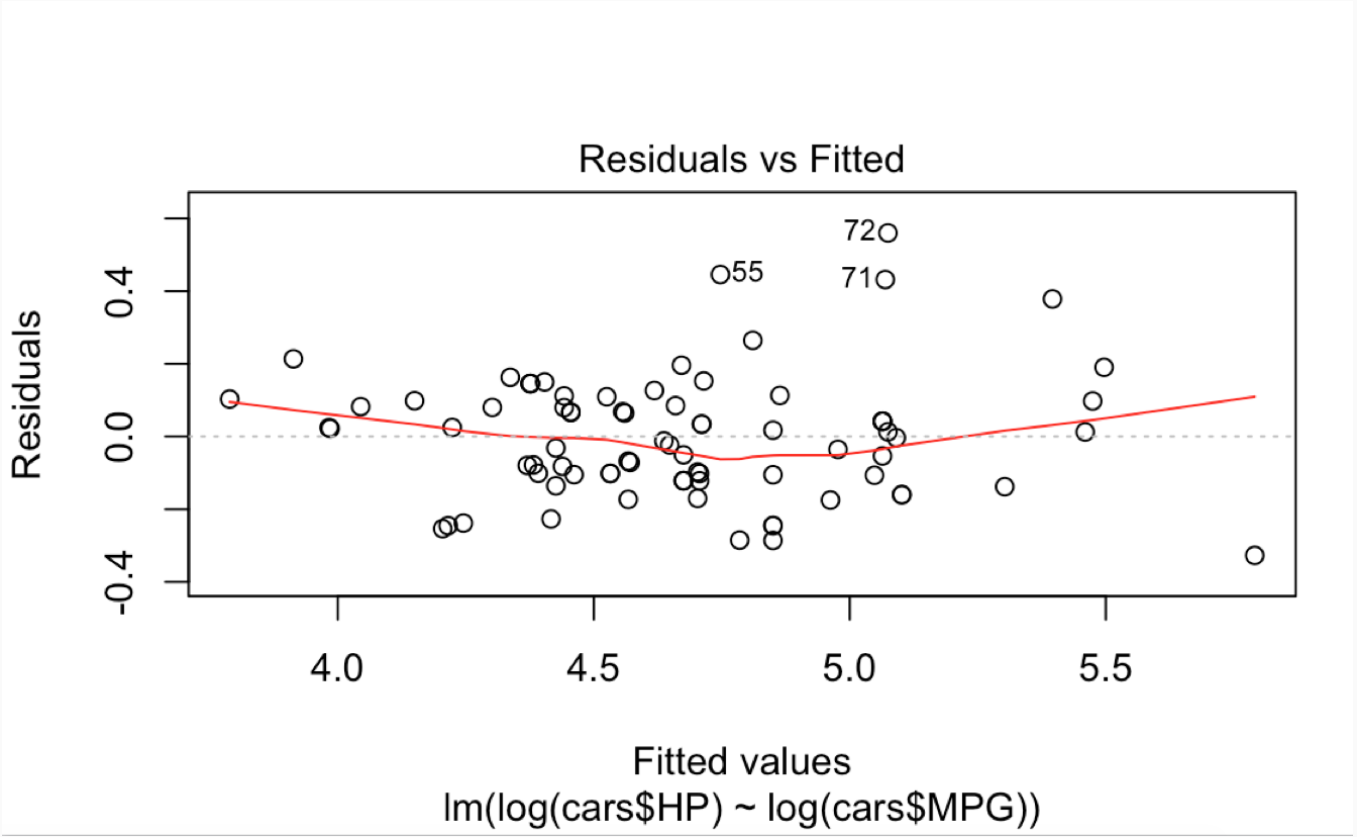
- Plot a histogram of the estimated errors (called **residuals**)
- QQplot
- Many tests exist: Kolomogorov-Smirnov, Shapiro-Wilk, ...



Assumptions: how do we check them?

Homoscedasticity

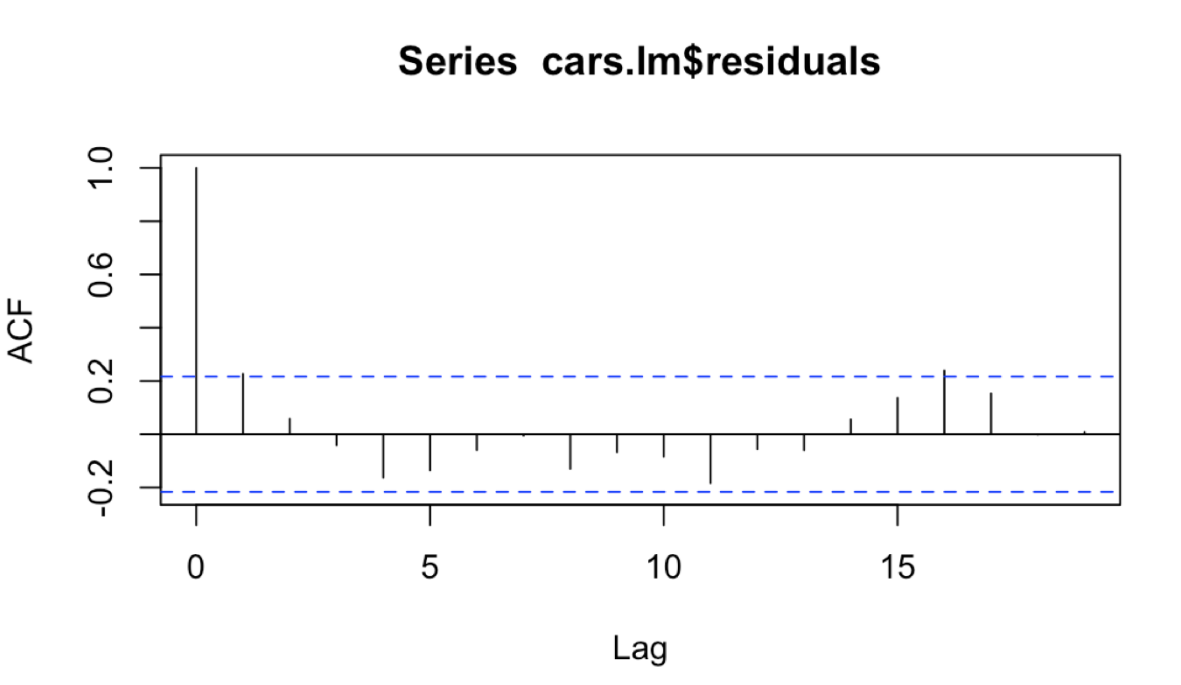
- Plot of Y vs. residuals should have equal variation across vertical slices
- Tests: Brusch-Pagan, White test, ...



Assumptions: how do we check them?

Independence of errors

- Autocorrelation plots
  - Most of the residuals should fall within the 95% confidence band around 0
- Durban-Watson test



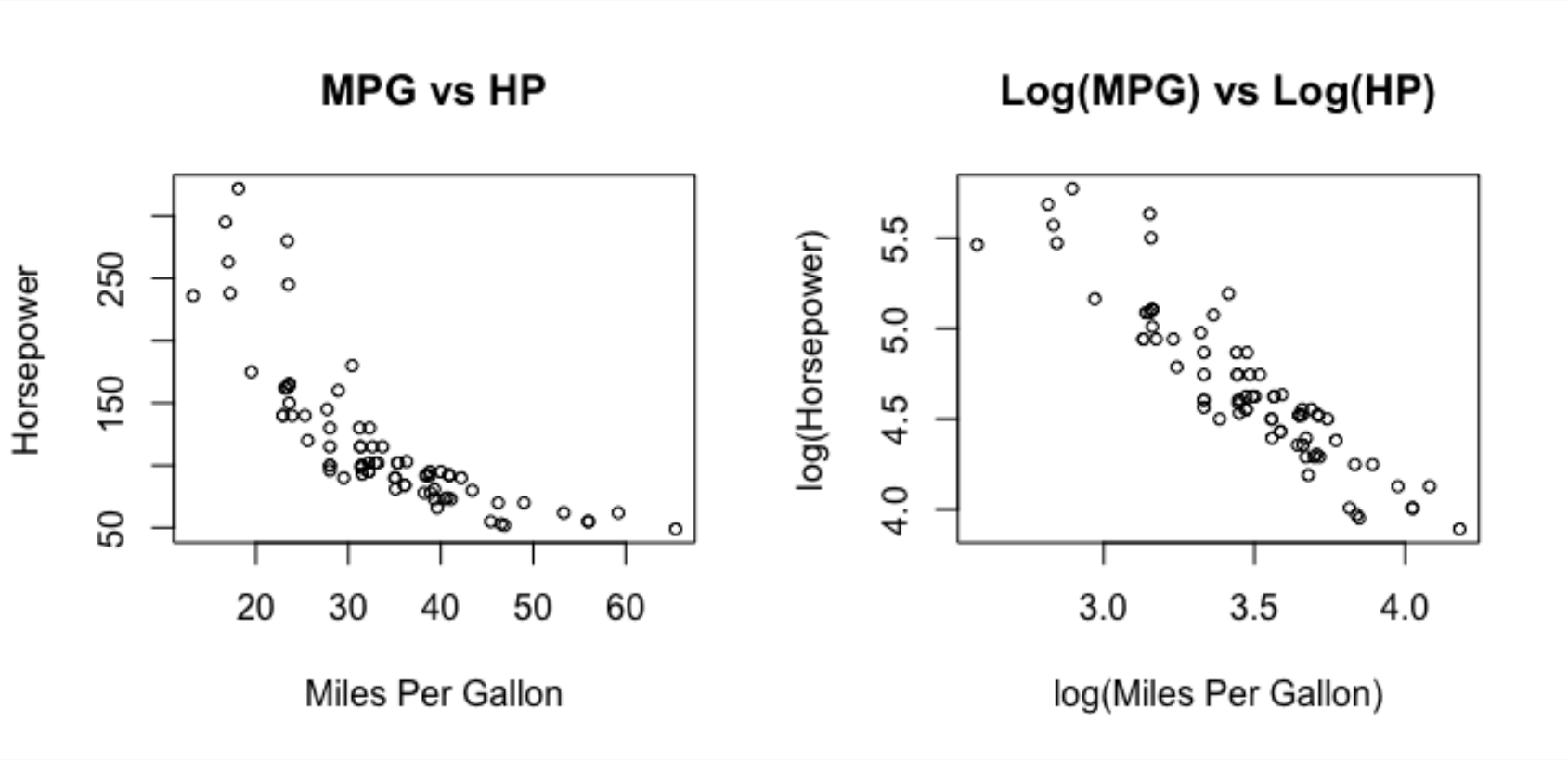
Assumptions: what do we do if they are not satisfied?

- If the data is nonlinear...
  - Try performing a transformation on the independent or dependent variables such as squaring it, taking the log or square root, or ...
- If the errors are not normal...
  - Often, this isn't a big problem
  - Transformations help here too
  - Maybe subsets of the data are more normal than the overall set
  - Outliers and/or high leverage points may contribute to this issue

Assumptions: what do we do if they are not satisfied?

- If the data exhibits heteroscedasticisity...
  - Log transformations are helpful
  - Search for and remove outliers/high-leverage points
  - Use a more advanced model (ARCH: auto-regressive conditional heteroscedasticity)
  - Hetereoscedasticity may arise from violation of one of the other assumptions
- If the errors are not independent...
  - You have a structural problem in your model
  - Very hard to fix...
  - One way that I am aware of: identify an appropriate ARMA process and fit a generalized least squares model

Example of the beauty of a log transform



Model Testing

Model Testing: Questions

Once we have estimated $\hat{\theta}$, we have some questions:

- Is θ_i significantly different from 0? (Is the variable X_i relevant?)
- How confident are we about what the true θ is?
- How do we know what independent variables to use?

Model Testing: Answers

Once we have estimated $\hat{\vec{\theta}}$, we have some questions:

- Is θ_i significantly different from 0? (Is the variable X_i relevant?)
  - Perform some hypothesis tests
  - t-tests, F-tests, etc...
    `https://en.wikipedia.org/wiki/Statistical_hypothesis_testing`
- How confident are we about what the true θ is?
  - Construct a confidence interval (many different kinds)
    `https://en.wikipedia.org/wiki/Confidence_interval`
- How do we know what independent variables to use?
  - Let's talk about this one some more

Feature Selection (Model Validation)

ML@B

- Before we do any feature selection, we need to make sure to split our dataset into a **training set** and a **validation set**
- Greedy forwards selection
- Greedy backwards selection
- Other search algorithms...
- Many different "goodness" metrics exist to compare models:
  - R^2 (want more), MSE (want less), AIC and BIC (want less), ...
  - MSE (mean squared error):

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

Next Steps

Generalizations of the linear model

ML@B

- **Classification problems**: logistic regression, support vector machines
- **Non-linearity**: kernel smoothing, splines and generalized additive models; nearest neighbor methods
- **Interactions**: tree-based methods, bagging, random forests and boosting (also capture non-linearities)
- **Regularized fitting**: ridge regression and lasso

Other related methods of interest for the practical data scientist...

ML@B

- Polynomial transformations
- Basis expansions
- Dummy coding of categorical inputs
- Time series models
- Hierarchical modeling
- Causal inference
- Spatial models

Questions?