

Linear Regression - Introduction

Objectives

After this lecture you should be able to:

- Define linear regression
 - Difference between simple and multiple linear regression
- Understand the differences between predictive and inferential linear regression
- Describe how coefficients in linear regression are determined
- Describe how to work with both continuous, categorical, engineered features
- Assess the performance of a linear regression model
- Interpret an Ordinary Least Squares linear regression summary from Statsmodels

Review Supervised vs. Unsupervised Learning

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs.

X, y -> predicting y based on the values in X

X, y a.k.a

features, target

independent, dependent

exogenous, endogenous

predictors, response

Example capstone: [Avalanche Prediction](#)

Unsupervised learning is a type of self-organized ... learning that helps find previously unknown patterns in data set without pre-existing labels.

X -> understanding structure in X

Example capstone: [Spice Blends](#)

-Wikipedia

Review Parametric vs Non-parametric models

A machine learning algorithm can be supervised or unsupervised, and parametric or non-parametric.

A **parametric** algorithm

- has a fixed number of parameters
- makes assumptions about the structure of the data
- will work well if the assumptions are correct!
- common examples: linear regression, neural networks, statistical distributions defined by a finite set of parameters

A **non-parametric** algorithm

- uses a flexible number of parameters, and the number of parameters often grows as it learns from more data.
- makes fewer assumptions about the data
- common examples: K-Nearest Neighbors, decision trees

Linear Regression

Given a data set $\{y_i, x_{i1}, \dots, x_{ip}\}_{i=1}^n$ of n statistical units, a linear regression model assumes that **the relationship between the dependent variable y and the p -vector of regressors x is linear.**

Assuming that there is only one predictor leads to modeling the relationship using **simple linear regression**:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

More than one predictor leads to **multiple linear regression**:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p + \epsilon$$

Y response

ϵ irreducible error

X predictors

β parameters/coefficients (what we need to figure out from *fitting* data)

Linear Regression

$$Y = \beta_0 + \beta_1 X + \epsilon$$

$$Y \approx \beta_0 + \beta_1 X$$

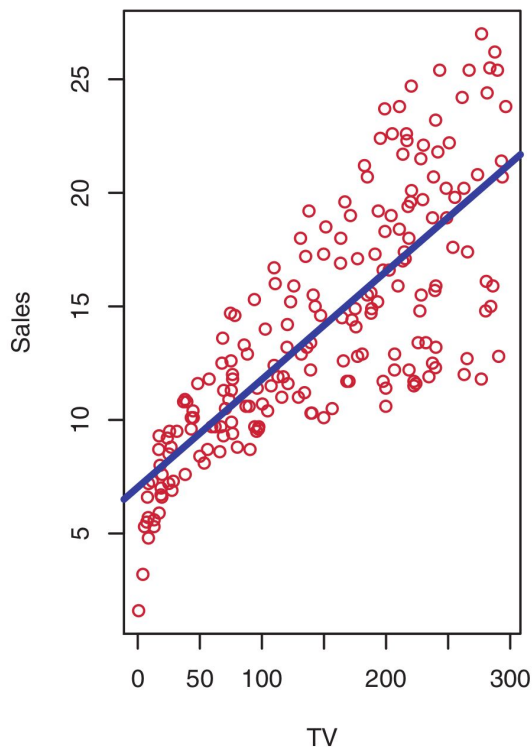
$$\text{sales} \approx \beta_0 + \beta_1 \times \text{TV}$$

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

$$\hat{y} = 7 + 0.05 x \quad \text{blue line} \rightarrow$$

^ estimated value of
unknown parameter,
predicted value

Sales as a $f(\text{TV advertising})$



Predictive Linear Regression

Goal:

Accurately predict a target

We picked a model based on trying different features, feature engineering, evaluation using cross validation.

We care that it predicts well on unseen data.

We don't really care that:

- Some of the features may be partially collinear (more later)
- Because of that, we can't rely on our parameter estimates to tell us something about their effect on the signal
- We may be violating some fundamental assumptions of inferential linear regression (more later)

Inferential Linear Regression

Goal:

Learn something accurate about the process that made the data. Infer (estimate) coefficients.

We picked a model based on trying different features, feature engineering, and checking residuals to see if we are violating some of the assumptions of linear regression.

We care that the parameter estimates are accurate and valid.

We don't really care that:

- It predicts well (but, it should!)

So what should you do?

It depends! (The typical data scientist answer)

Most often, we're asked for predictive models, **however**, one major benefit to linear regression is interpretability of the coefficients. If assumptions of inferential linear regression are being violated, then you can't rely on your parameter estimates to provide interpretability.

It is wise to take a hybrid approach: predictive (best model through cross validation) but attempt to fulfill the assumptions of inferential linear regression so that you have confidence in your parameter estimates.

Linear Regression

$$Y = \beta_0 + \beta_1 X + \epsilon$$

$$Y \approx \beta_0 + \beta_1 X$$

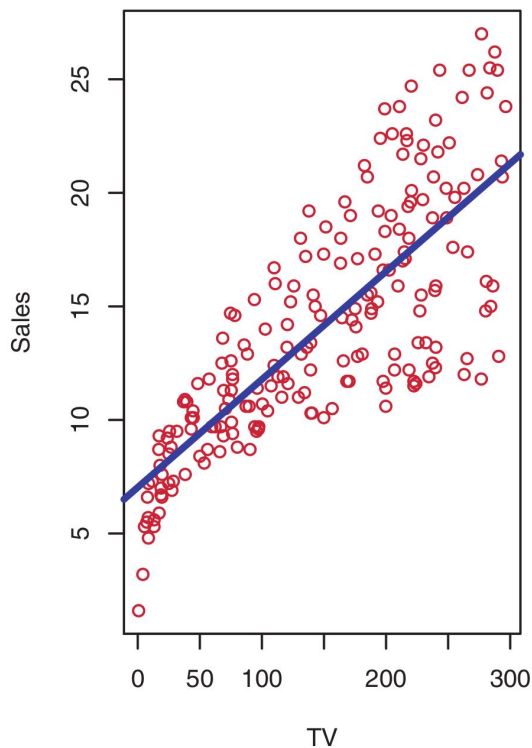
$$\text{sales} \approx \beta_0 + \beta_1 \times \text{TV}$$

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

$$\hat{y} = 7 + 0.05 x \quad \text{blue line} \rightarrow$$

- ^ estimated value of unknown parameter, predicted value

Sales as a $f(\text{TV advertising})$



Interpreting linear regression coefficients

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

How to interpret $\hat{\beta}_1$?

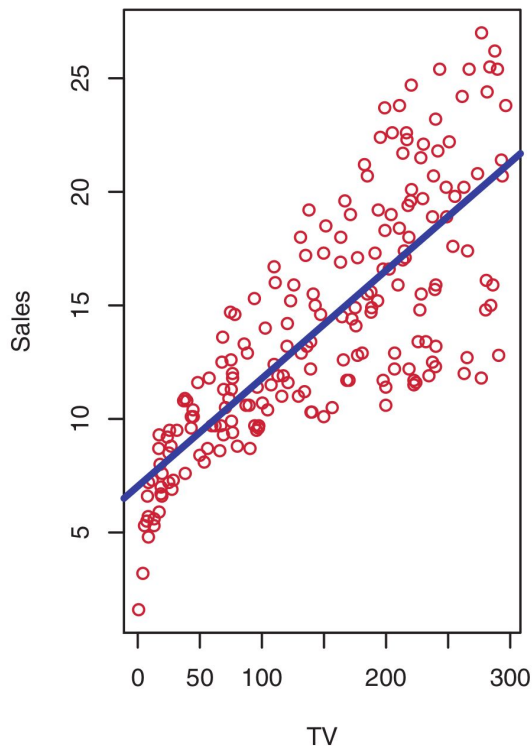
Holding all else constant, a 1 unit increase in x_1 increases the response \hat{y} by $\hat{\beta}_1$.

$$\hat{y} = 7 + 0.05 x$$

$$\text{sales} \approx \beta_0 + \beta_1 \times \text{TV}$$

Increasing TV by 1 increases sales by 0.05.

Sales as a $f(\text{TV advertising})$



Determining the coefficients (*fitting* the model)

The coefficients β are unknown. We'd like to find values for them so that the resulting prediction (line in simple regression, multidimensional plane in multiple regression) is as close to the training data as possible.

This is a [mathematical optimization](#) problem.

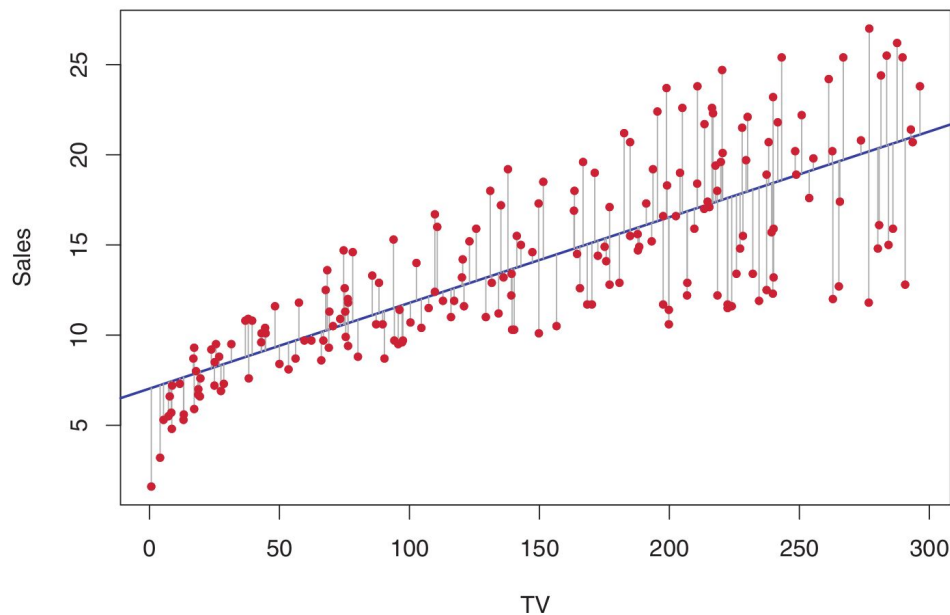
In mathematical optimization, we need an objective function to maximize or minimize.

In machine learning, we typically talk about making loss/cost functions. The goal of the optimization is to find the parameters/coefficients/weights that minimize the cost/loss function.

Cost function: Residual Sum of Squares (RSS)

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

$y_i - \hat{y}_i$ is a residual



Mean squared error is the average RSS:

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

Least squares: solve β by minimizing squared residuals

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Any given set of parameters/coefficients will give a value of RSS. We'd like to find the set that give the minimum RSS.

This will be the minimum of the cost function (a surface in multidimensional space).

If the cost function is simple enough, you can take its derivative, set it equal to zero, and solve for the coefficients -> yields an analytic solution for the coefficients. (Great, this is fast!)

If the cost function is more complex (very often the case), resort to numerical optimization. Commonly, gradient descent (more later).

Least squares analytic solution for linear regression

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

See [derivation](#) again.

Make matrices clear

Design Matrix \mathbf{X} :

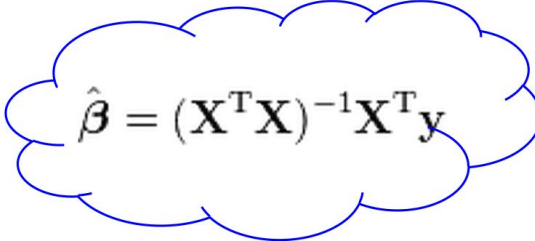
$$\mathbf{X} = \begin{bmatrix} 1 & X_{1,1} & X_{1,2} & \cdots & X_{1,p-1} \\ 1 & X_{2,1} & X_{2,2} & \cdots & X_{2,p-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n,1} & X_{n,2} & \cdots & X_{n,p-1} \end{bmatrix}$$

Target:

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Coefficient matrix β :

$$\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \end{bmatrix}$$


$$\hat{\beta} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

Independent Variables in Regression

There are two types of independent variables in the feature matrix:

1. Continuous Variables
2. Categorical Variables

Handling continuous features

There are two ways to deal with continuous variables

Leave as is (the standard)	Standardize/Transform (if necessary)
Great for interpretation, we all know what a one year increase in age is!	Harder for lay-people to understand what a one unit increase in a standardization unit
Might not hold true with some of the assumptions	Transforming the data can help with upholding the assumptions
	Standardization of independent variables allows for comparisons of estimated coefficients on the dependent variable, not importance of the variable

Handling categorical features

Binary encoding (yes/no, true/false, exists or not).

Example: Sex (Male/Female)

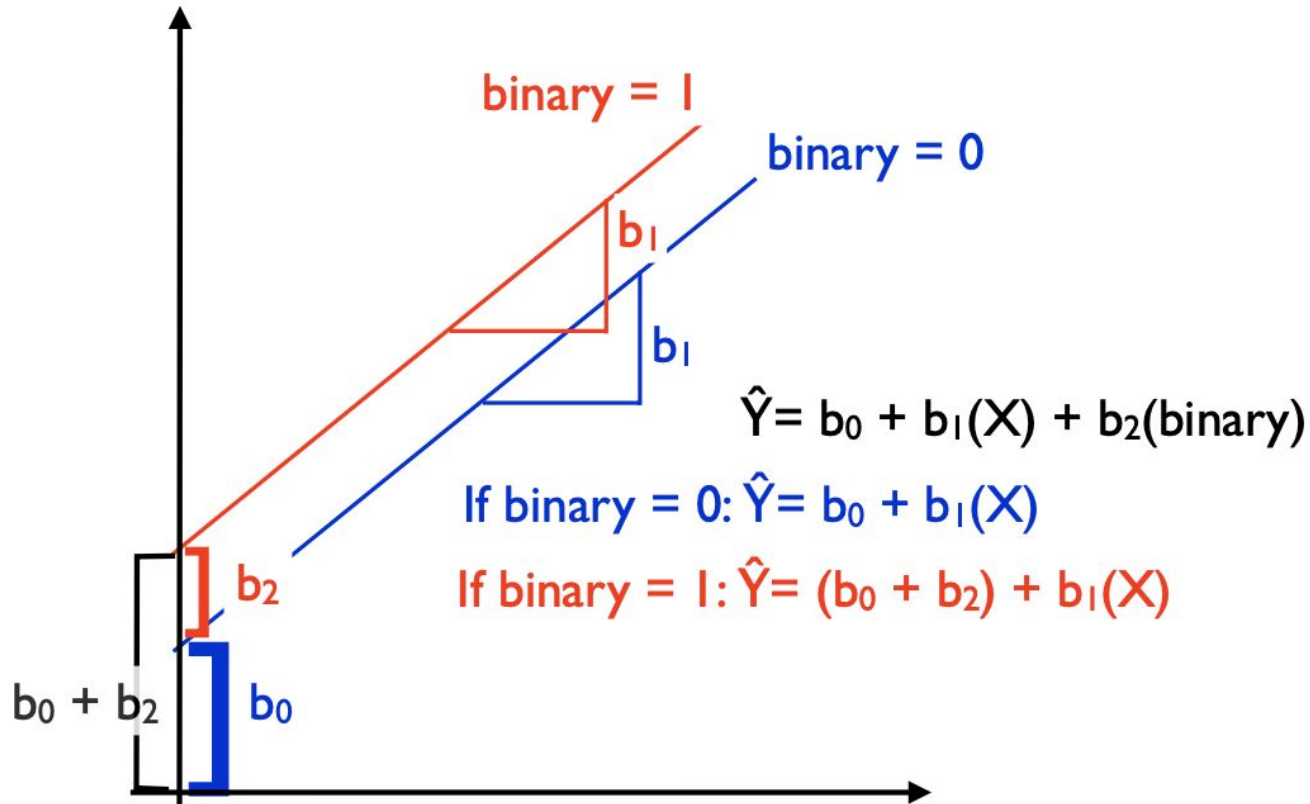
$$x_i = \begin{cases} 1 & \text{if } i\text{th person is female} \\ 0 & \text{if } i\text{th person is male} \end{cases}$$

$$y_i = \beta_0 + \beta_1 \underline{x_i} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is female} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is male.} \end{cases}$$

β_1 quantifies how much being female changes the response relative to male.

We call the group encoded as 0 the “reference group”

Independent Variables: Categorical - Binary



Handling categorical features

Multi-category encoding

Example: Ethnicity (Asian/Caucasian/African American)

$$x_{i1} = \begin{cases} 1 & \text{if } i\text{th person is Asian} \\ 0 & \text{if } i\text{th person is not Asian} \end{cases}$$

$$x_{i2} = \begin{cases} 1 & \text{if } i\text{th person is Caucasian} \\ 0 & \text{if } i\text{th person is not Caucasian} \end{cases}$$

$$y_i = \beta_0 + \beta_1 \underline{x_{i1}} + \beta_2 \underline{x_{i2}} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is Asian} \\ \beta_0 + \beta_2 + \epsilon_i & \text{if } i\text{th person is Caucasian} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is AA.} \end{cases}$$

β_1 quantifies how much being Caucasian changes the response relative to African American.

β_2 quantifies how much being Asian changes the response relative to being African American

Our reference group here is African American, but how did this get coded?

Handling categorical features

Multi-category encoding

Example: Ethnicity (Asian/Caucasian/African American)

$$x_{i1} = \begin{cases} 1 & \text{if } i\text{th person is Asian} \\ 0 & \text{if } i\text{th person is not Asian} \end{cases}$$

$$x_{i2} = \begin{cases} 1 & \text{if } i\text{th person is Caucasian} \\ 0 & \text{if } i\text{th person is not Caucasian} \end{cases}$$

$$y_i = \beta_0 + \beta_1 \underline{x_{i1}} + \beta_2 \underline{x_{i2}} + \epsilon_i = \begin{cases} \beta_0 + \beta_1 + \epsilon_i & \text{if } i\text{th person is Asian} \\ \beta_0 + \beta_2 + \epsilon_i & \text{if } i\text{th person is Caucasian} \\ \beta_0 + \epsilon_i & \text{if } i\text{th person is AA.} \end{cases}$$

Data

Ones	Ethnicity
1	AA
1	Asian

Recode Design Matrix

Ones	Asian	Caucasian
1	0	0
1	1	0

Handling categorical features - Multilevel

For multi-level variables we have to create **dummy variables** ([in Python](#)) by a process called [one-hot encoding](#). A dummy variable is an indicator (1/0) variable which indicates one level out of k-categories.

id	color
1	red
2	blue
3	green
4	blue

One-Hot
Encoding

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Handling categorical features - Multilevel

For multi-level variables we have to create **dummy variables** ([in Python](#)) by a process called [one-hot encoding](#). A dummy variable is an indicator (1/0) variable which indicates one level out of k-categories.

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Now, to pick a reference group, aka the column you drop from your dataset and the column that you want all the other groups to be compared to. Typically the majority group is dropped or if there is some sort of ordinality to your category levels (first, second, third), chose the group you want to compare to. This is subjective and up to you. **Note: Best practice is that you do not compare non-reference groups to each other, though with a little algebra, it is possible**

Handling categorical features - Multilevel

For multi-level variables we have to create **dummy variables** ([in Python](#)) by a process called [one-hot encoding](#). A dummy variable is an indicator (1/0) variable which indicates one level out of k-categories.

id	color_red	color_green
1	1	0
2	0	0
3	0	1
4	0	0

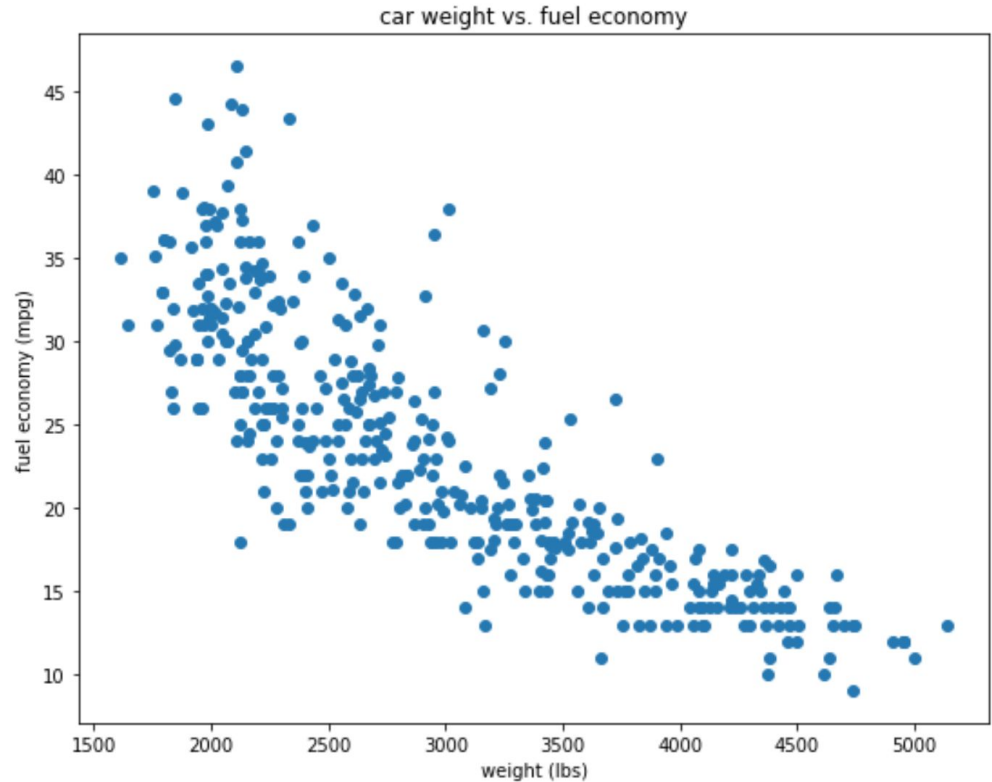
We picked blue, as in this case it was the majority. Now the regression model will include color_red and color_green, each compared to the reference group, color_blue.

For k-level categorical independent variables, we need to create k-1 dummy variables!

Engineered features: making curves out of straight lines

The relationship between weight and fuel economy is not a straight line.

This makes sense if you think about it: the fuel economy of a car can never be negative - it will go to zero as the weight increases.



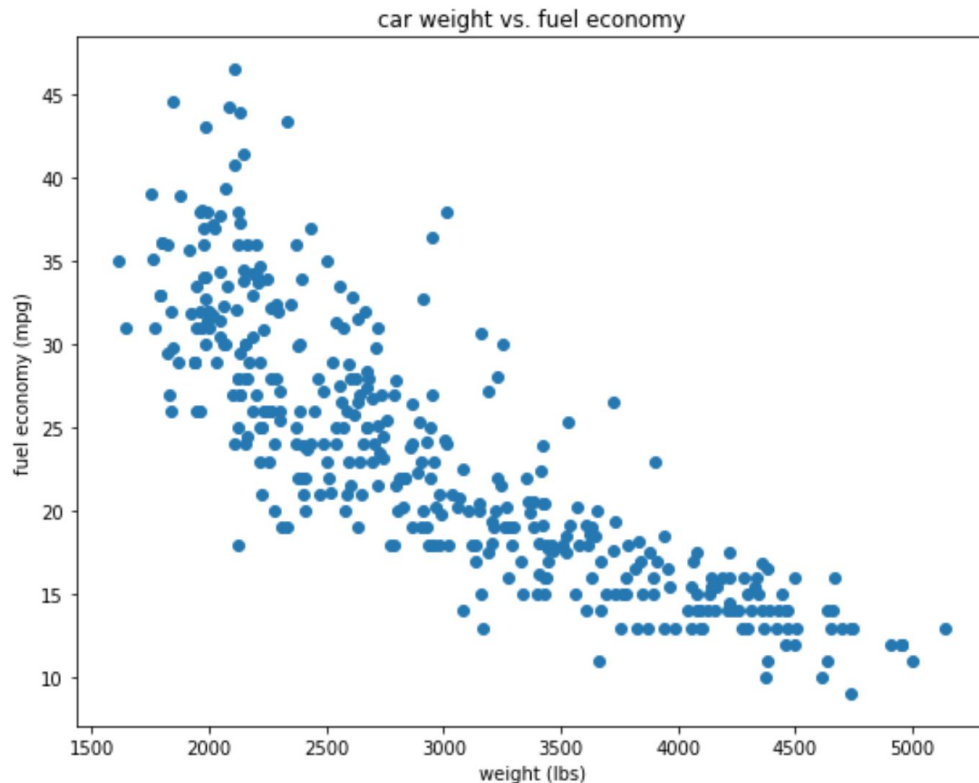
Engineered features: making curves out of straight lines

One solution is to perform linear regression on engineered features.

One potential model:

$$y = \beta_0 + \frac{\beta_1}{x_{weight}}$$

might seem nonlinear, but it's simply a linear function of the predictor $\frac{1}{x_{weight}}$



Engineered features: making curves out of straight lines

One popular strategy for fitting complex curves is polynomial regression, which involves creating polynomial features from observed features.

We will take a look at an example in a bit.



Assessing the performance/usefulness of your model

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Residual sum of squares - ok, but depends on n

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

Mean squared error - better, but in squared units of response

$$\text{RMSD}(\hat{\theta}) = \sqrt{\text{MSE}(\hat{\theta})}$$

Root mean square error (or deviation) - in units of response (similar to RSE previous slide)

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}}$$

R squared, ranges from 0 to 1 (but doesn't penalize for model complexity), where $\text{TSS} = \sum (y_i - \bar{y})^2$

$$\bar{R}^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1}$$

Adjusted R-squared (0 to 1), penalizes for model complexity according to number of predictors, p

Assessing the performance/usefulness of your model

$$\text{RSS} = \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Residual sum of squares - ok, but depends on n

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

Mean squared error - better, but in squared units of response

$$\text{RMSD}(\hat{\theta}) = \sqrt{\text{MSE}(\hat{\theta})}$$

Root mean square error (or deviation) - in units of response

$$R^2 = \frac{\text{TSS} - \text{RSS}}{\text{TSS}} = 1 - \frac{\text{RSS}}{\text{TSS}}$$

R squared, ranges from 0 to 1 (but doesn't penalize for model complexity), where $\text{TSS} = \sum (y_i - \bar{y})^2$

$$\bar{R}^2 = 1 - (1 - R^2) \frac{n - 1}{n - p - 1}$$

Adjusted R-squared (0 to 1), penalizes for model complexity according to number of predictors, p

Walking through linear regression

`predictive-linear-regression.ipynb`

Assessing the performance/usefulness of your model

There were hypothesis tests checking if coefficients in the model were significant (not zero).

The F-statistic is a hypothesis test where the Null Hypothesis is that **all the coefficients are = 0** (so model is not useful).

$$H_0 : \beta_1 = \beta_2 = \dots = \beta_p = 0$$

The alternative is that at least one of the coefficients is not 0.

$$H_a : \text{at least one } \beta_j \text{ is non-zero.}$$

$$F = \frac{(\text{TSS} - \text{RSS})/p}{\text{RSS}/(n - p - 1)}$$

For no relationship between predictors and response (H_0), F statistic is near 1.

Interpretation of OLS summary from Statsmodels.

```

=====
                        OLS Regression Results
=====
Dep. Variable:          y      R-squared:          0.933
Model:                  OLS    Adj. R-squared:     0.928
Method:                 Least Squares    F-statistic:    211.8
Date:                   Mon, 03 Nov 2014    Prob (F-statistic): 6.30e-27
Time:                   14:45:06    Log-Likelihood:  -34.438
No. Observations:      50    AIC:              76.88
Df Residuals:          46    BIC:              84.52
Df Model:               3
Covariance Type:        nonrobust
=====

```

	coef	std err	t	P> t	[95.0% Conf. Int.]	
x1	0.4687	0.026	17.751	0.000	0.416	0.522
x2	0.4836	0.104	4.659	0.000	0.275	0.693
x3	-0.0174	0.002	-7.507	0.000	-0.022	-0.013
const	5.2058	0.171	30.405	0.000	4.861	5.550

```

=====
Omnibus:                0.655    Durbin-Watson:          2.896
Prob(Omnibus):           0.721    Jarque-Bera (JB):         0.360
Skew:                    0.207    Prob(JB):                 0.833
Kurtosis:                3.026    Cond. No.:                 221.
=====

```

Proportion of Variance Explained by model is 93.3%

Measure of the significance of the fit ...my model isn't utterly useless 😊

There is an approximately 95% chance that [0.275, 0.693] will contain the true value of β_2

Each coefficient is really significant. Can also think of this as a Partial F-test.

“The average effect on Y of a one unit increase in X₂, holding all other predictors (X₁ & X₃) fixed, is 0.4836”

- However, interpretations are generally pretty hazardous due to correlations among predictors.
- p-values for each coefficient ≈ 0 , so might be okay here

Note: Magnitude of the Beta coefficients is NOT how to determine whether predictor contributes. Why?

Objectives

After this lecture you should be able to:

- Define linear regression
 - Difference between simple and multiple linear regression
- Understand the differences between predictive and inferential linear regression
- Interpret linear regression coefficients
- Describe how coefficients in linear regression are determined
- Describe how to work with both continuous, categorical, engineered features
- Assess the performance of a linear regression model
- Interpret an Ordinary Least Squares linear regression summary from Statsmodels