Linear Regression

Delta Analytics builds technical capacity around the world.

This course content is being actively developed by Delta Analytics, a 501(c)3 Bay Area nonprofit that aims to empower communities to leverage their data for good.

Please reach out with any questions or feedback to inquiry@deltanalytics.org.

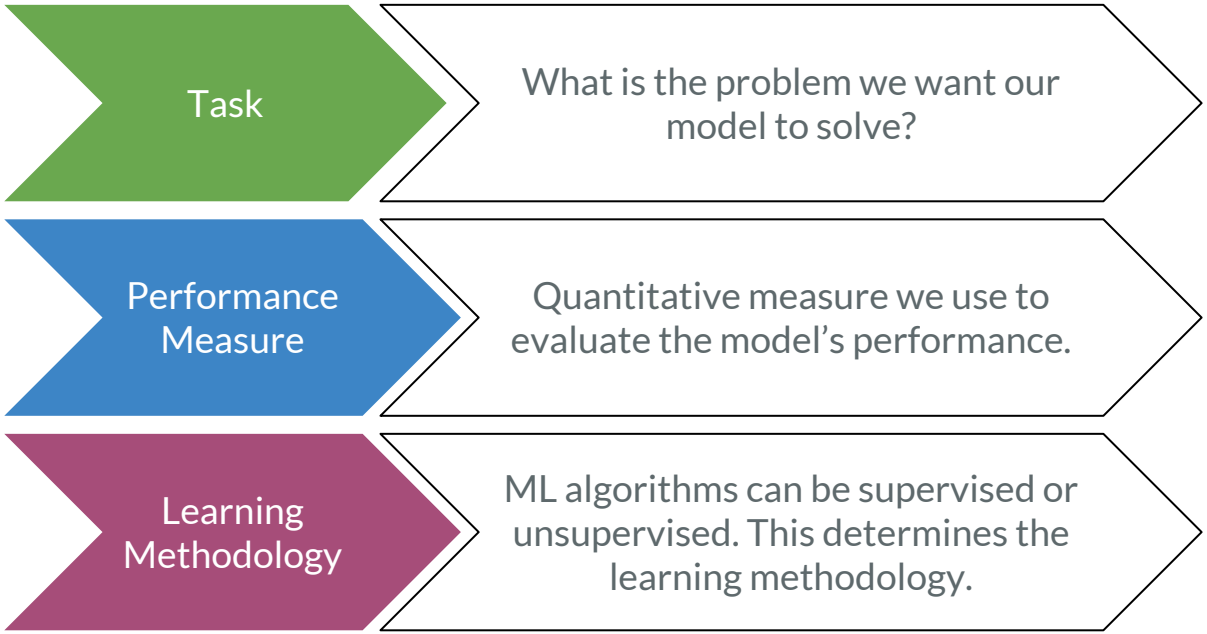
Find out more about our mission here.

Module 3:
Linear Regression

Let’s do a quick review of module 2!

How does a model learn from raw data?

All models have the following components:



Source: Deep Learning Book - Chapter 5: Introduction to Machine Learning

Exercise 1

What are the explanatory features? What is the outcome feature?

Classification task:

Outstanding Debt	Current Annual Income ($)	Approval for credit card Y	Predicted Approval for credit card Y*
200	12,000	No	Yes
60,000	60,000	No	No
0	11,000	Yes	No
10,000	200,000	Yes	Yes

Let's fill in the blanks for our credit approval example:

Task	
Performance Measure	
Learning Experience	

Let's fill in the blanks for our credit approval example:

Task	Should this customer be approved for a credit card?
Performance Measure	Log Loss
Learning Experience	Supervised

Exercise 2

What are the explanatory features? What is the outcome feature?

Regression task:

Time spent A week studying machine learning	Accuracy of classification model built by student	Predicted Accuracy of classification model
X	Y	Y*
10	90%	30%
2	30%	60%
12	95%	26%
0	50%	88%

Let's fill in the blanks for our studying example:

Task	
Performance Measure	
Learning Experience	

Let's fill in the blanks for our studying example:

Task	How accurate is this student's classification model?
Performance Measure	MSE
Learning Experience	Supervised

Module 3:
Linear Regression

Course overview:

- ✓ Module 1: Introduction to Machine Learning
- ✓ Module 2: Machine Learning Deep Dive
- ✓ Module 3: Linear Regression
- ❑ Module 4: Decision Trees
- ❑ Module 5: Ensemble Algorithms
- ❑ Module 6: Unsupervised Learning Algorithms
- ❑ Module 7: Natural Language Processing Part 1
- ❑ Module 8: Natural Language Processing Part 2

Now let's turn to the data we will be using...

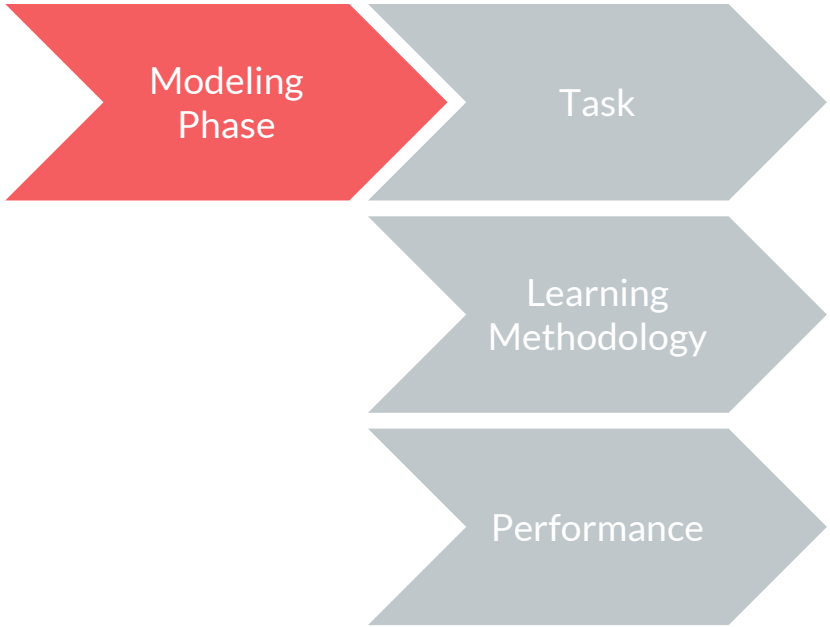
Module Checklist

- ❏ Linear regression
  - ❏ Relationship between two variables (x and y)
    - ❏ Formalizing f(x)
    - ❏ Correlation between two variables
    - ❏ Assumptions
  - ❏ Feature engineering and selection
  - ❏ Learning process: Loss function and Mean Squared Error
  - ❏ Univariate regression, Multivariate regression
  - ❏ Measures of performance (R2, Adjusted R2, MSE)
  - ❏ Overfitting, Underfitting

What is linear regression?

Linear regression is a model that explains the relationship between explanatory features and an outcome feature as a **line in two dimensional space**.

Why is linear regression important?



Linear regression has been in use since the 19th century & is **one of the most important machine learning tools** available to researchers.

Many other models are built upon the logic of linear models. For example, the most simple form of deep learning model, with no hidden layers, is a linear model!

Source: History of Linear Regression - Introduction to Linear Regression

Linear regression: model cheat sheet

Pros

- Very popular model with intuitive, easy to understand results.
- Natural extension of correlation analysis

Cons

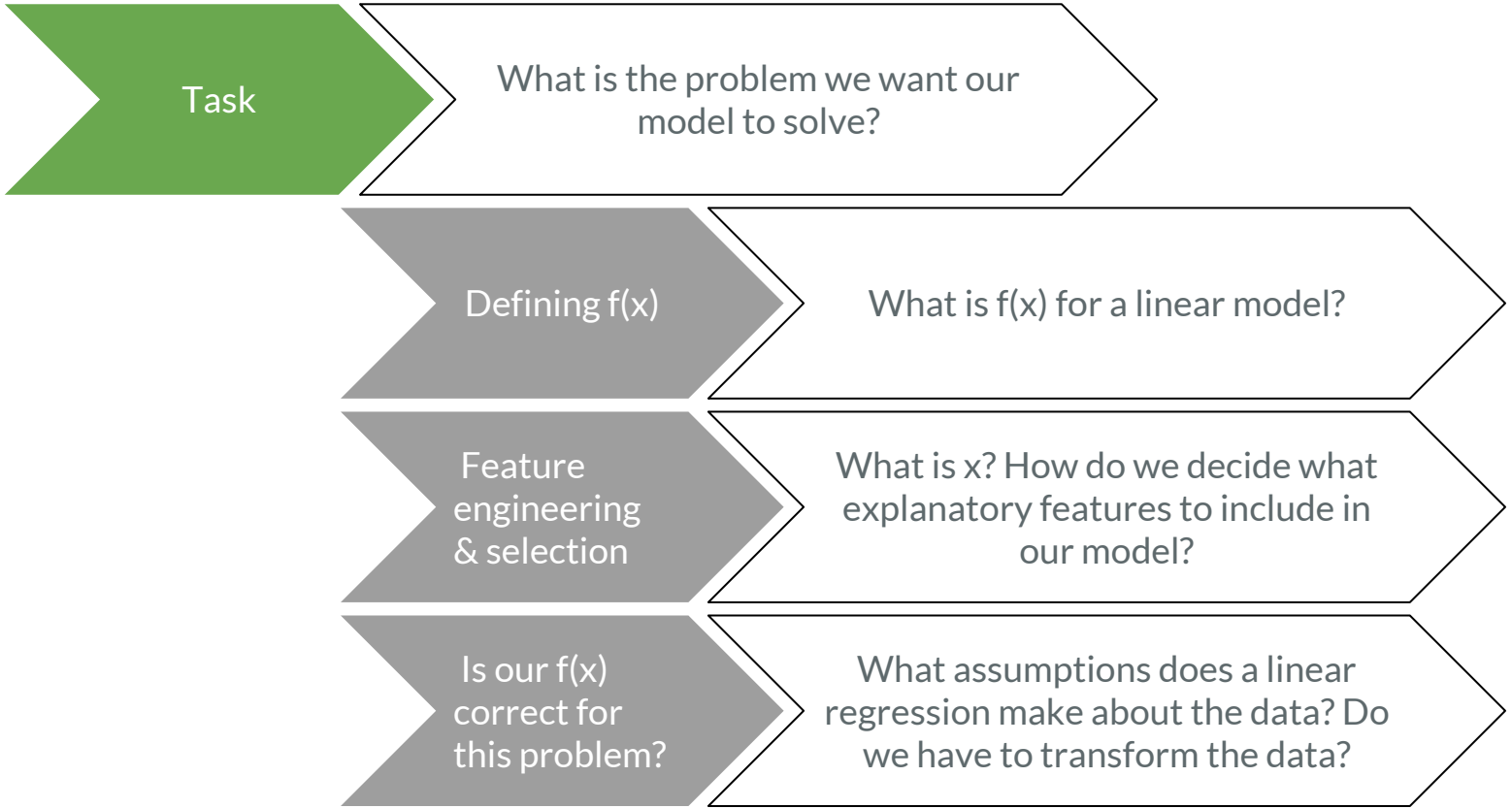
- Sensitive to outliers
- The world is not always linear; we often want to model more complex relationships
- Does not allow us to model interactions between explanatory features (we will be able to do this using a decision tree)

Assumptions*

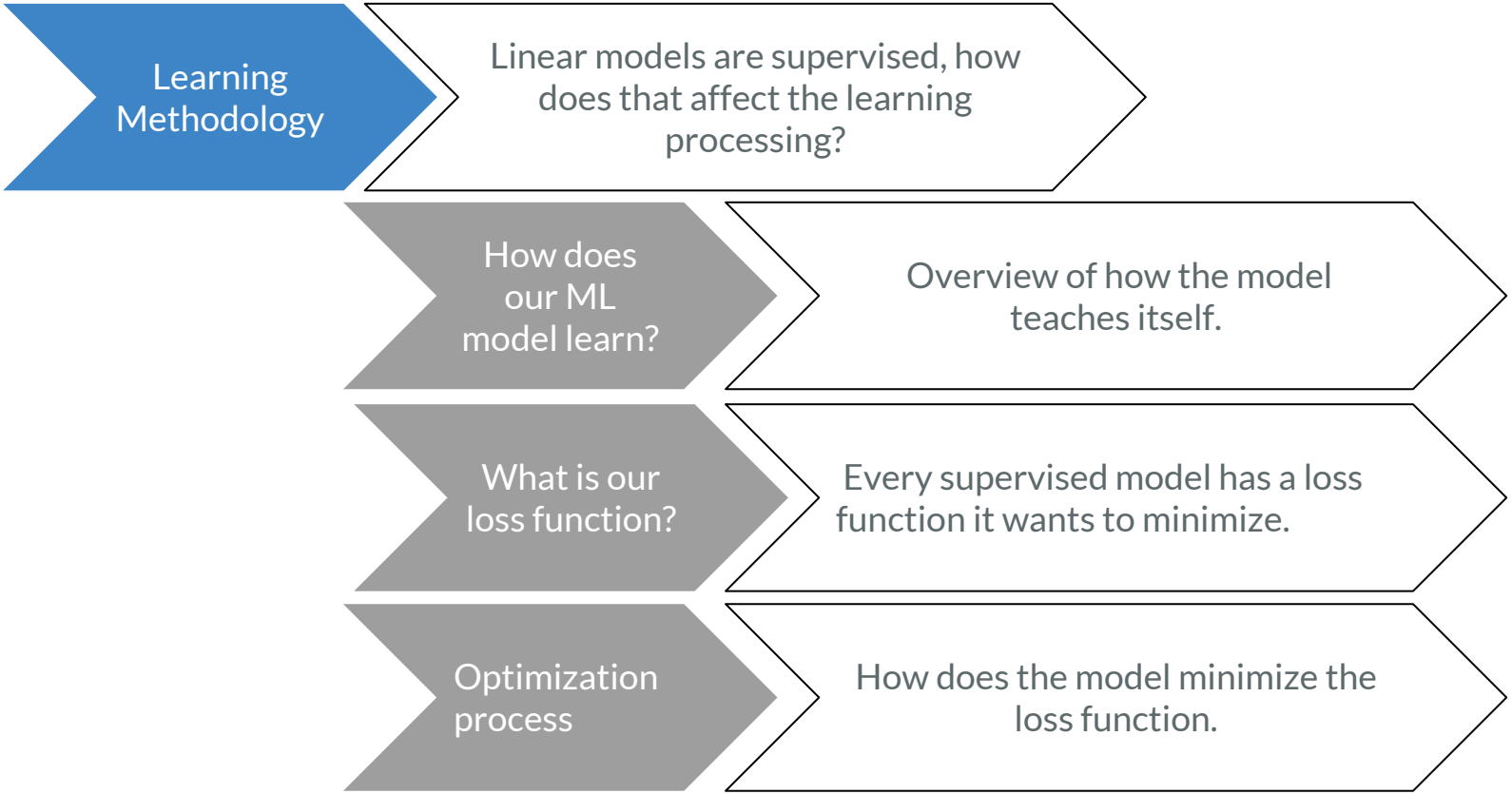
- Linear relationship between x and y
- Normal distribution of variables
- No multicollinearity (Independent variables)
- Homoscedasticity
- *Rule of thumb:* at least 20 observations per independent variable in the analysis

* We will explain each of these assumptions in this module!

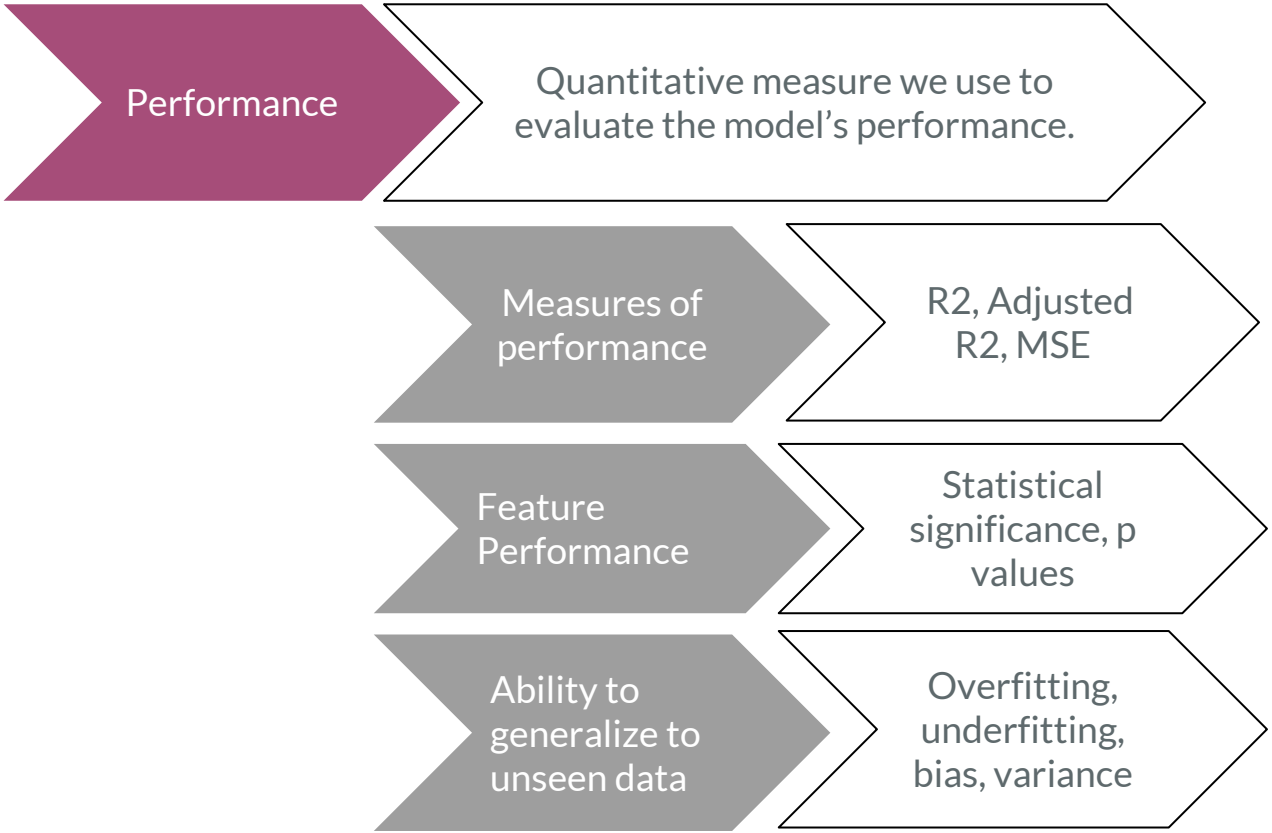
Today we are looking closer at each component of the framework we discussed in the last class



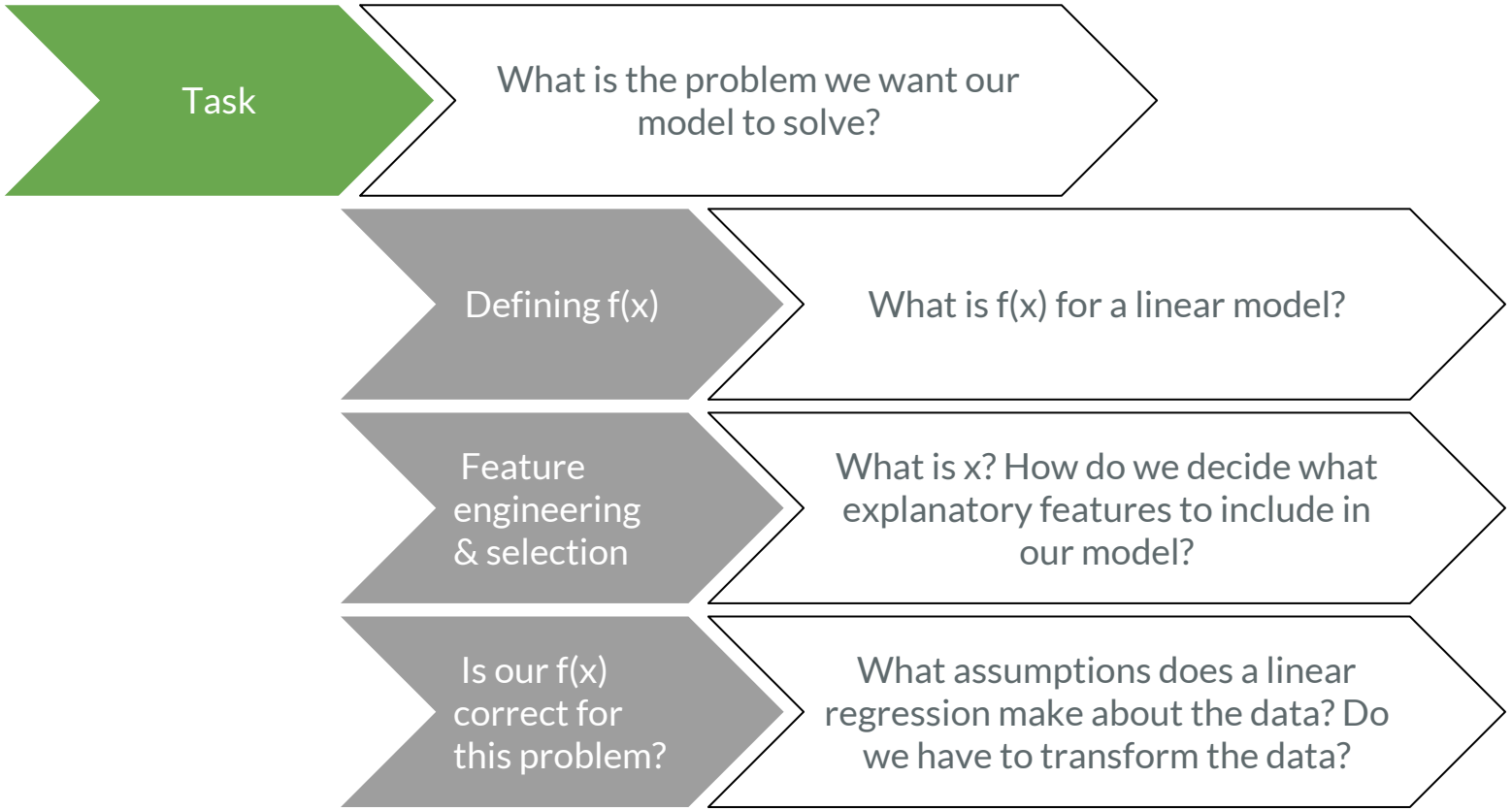
Learning methodology is how the linear regression model learns which line best fits the raw data.



There are linear models for both regression and classification problems.



The task


Task
What is the problem we want our model to solve?
Defining f(x)
What is f(x) for a linear model?
Feature engineering & selection
What is x? How do we decide what explanatory features to include in our model?
Is our f(x) correct for this problem?
What assumptions does a linear regression make about the data? Do we have to transform the data?

Task

Recall our discussion of two types of supervised tasks, regression & classification. There are linear models for both. Today we will only discuss regression.

Regression

Continuous variable

Ordinary Least Squares (OLS) regression

OLS is a linear regression method based on minimizing the sum of squared residuals

Classification

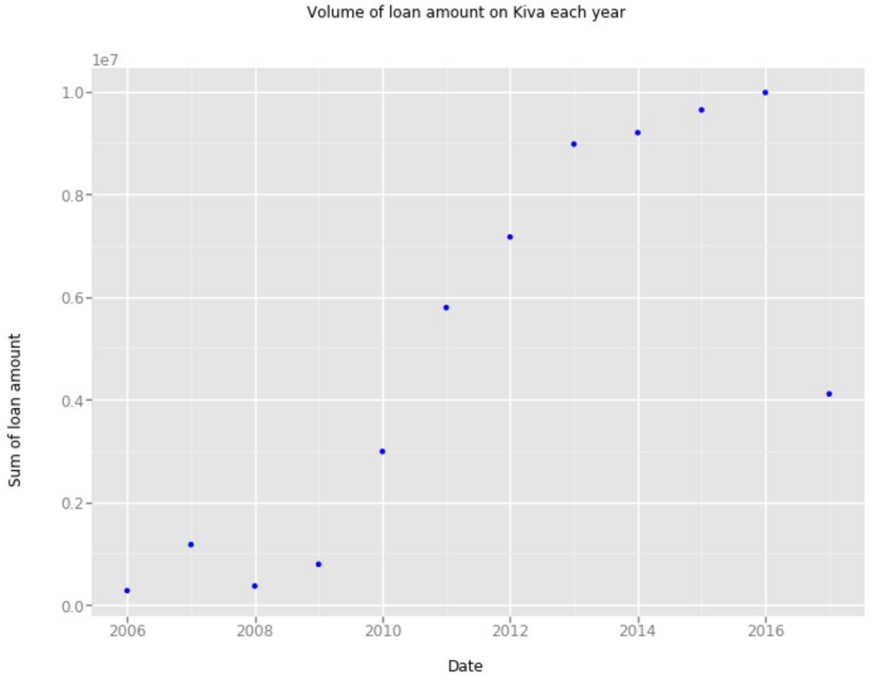
Categorical variable

Logistic regression

We will not cover this in the lecture, but we will provide resources for further study

OLS Regression Task

A OLS regression is a trend line that predicts how much Y will change for a given change in x. Let's try and break that down a little more by looking at an example.



We have plotted the total $ loaned by KIVA in Kenya each year. What is the trend? How would you summarize this trend in one sentence?

Human Intuition

"Every year, the value of loans in KE appears to be increasing"

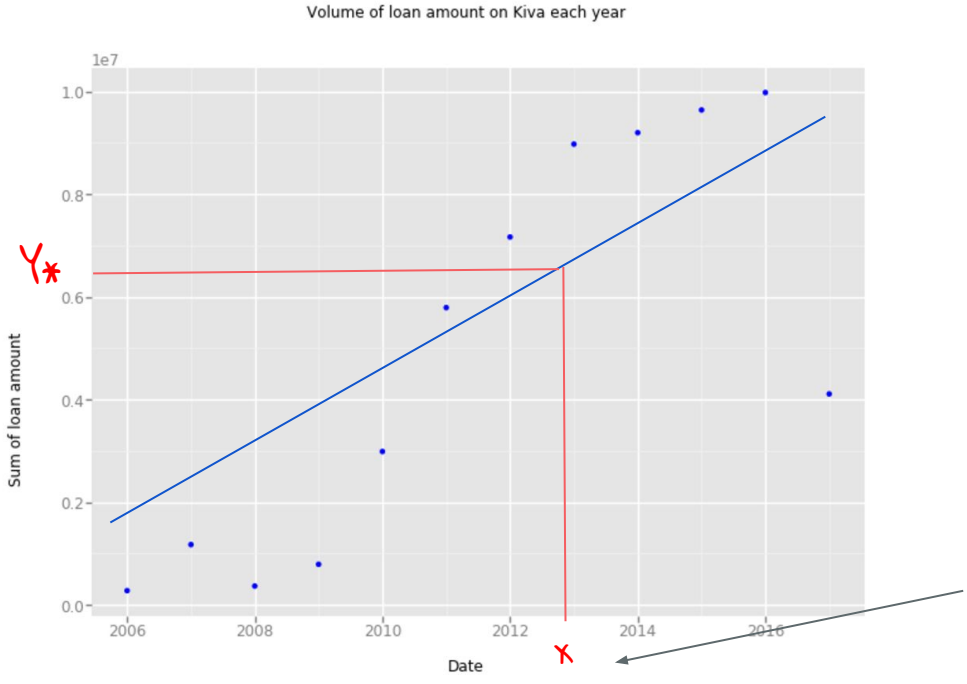
OLS Regression

Every additional year corresponds to x additional dollars Kiva loans in KE.

A linear regression formalizes our perceived trend as a relationship between x and Y. It can be understood intuitively as a trend line.

Defining f(x)

A linear model expresses the relationship between our explanatory features and our outcome features as a straight line. The output of f(x) for a linear model will always be a line.



A linear model allows us to predict beyond our set of observations because for every point on the x axis we can find the corresponding Y*

For example, we can now say for an x not in our scatterplot (May, 2012) what we predict the $ amount of loans is.

Is linear regression the right model for our data?

OLS Regression Task

Is our f(x) correct for this problem?

A big part of being a machine learning researcher involves choosing the right model for the task.

Each model makes certain assumptions about the underlying data.

Let’s take a closer look at how this relates in linear regression.

OLS Regression Task

Is our f(x) correct for this problem?

Before we choose a linear model, we need to make sure all the assumptions hold true in our data.

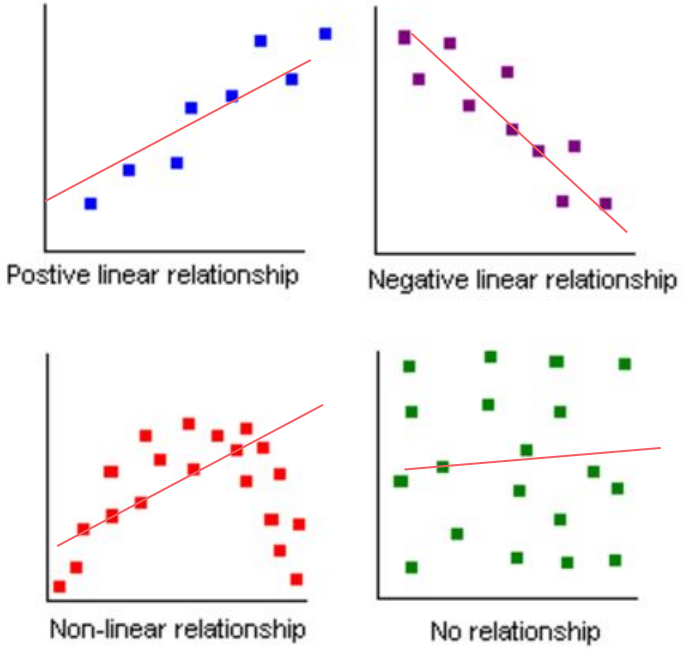
OLS Linear Regression Assumptions

- ❑ Linear relationship between x and Y
- ❑ Normal distribution of variables
- ❑ No autocorrelation (Independent variables)
- ❑ Homoscedasticity
- ❑ No multicollinearity
- ❑ Rule of thumb: at least 20 observations per independent variable in the analysis

OLS Regression Task

Is our f(x) correct for this problem?

Is there a linear relationship between x and Y?



Linear regression assumes that there is a linear relationship between x and Y.

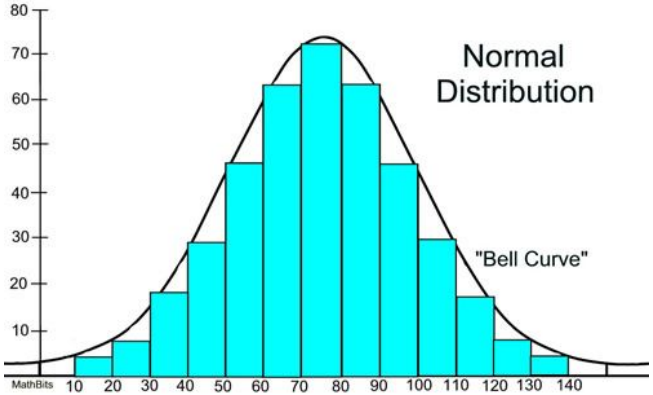
If this is not true, our trend line will do a poor job of predicting Y.

Source: Linear and non-linear relationships

OLS Regression Task

Is our f(x) correct for this problem?

Is our data normally distributed?



Normal distribution of explanatory and outcome features avoids distortion of results due to outliers or skewed data.

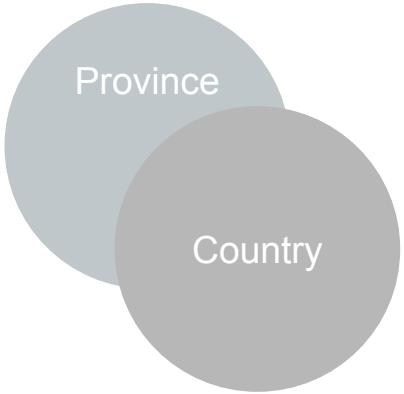
Source: Statistics - Normal distribution

OLS Regression Task

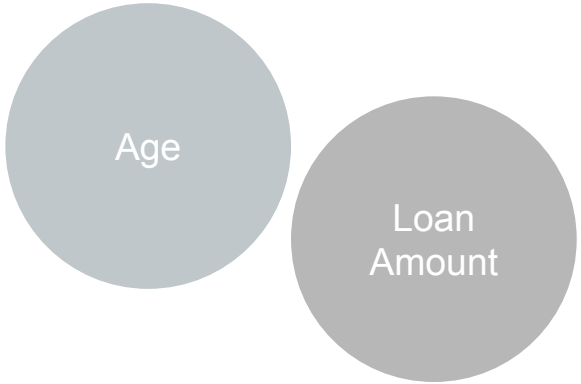
Is our f(x) correct for this problem?

Multicollinearity occurs when explanatory variables are highly correlated. Do we have multicollinearity?

Multicollinearity introduces redundancy to the model and reduces our certainty in the results. We want no multicollinearity in our model.



Province and county are highly correlated. We will want to include only one of these variables in our model.

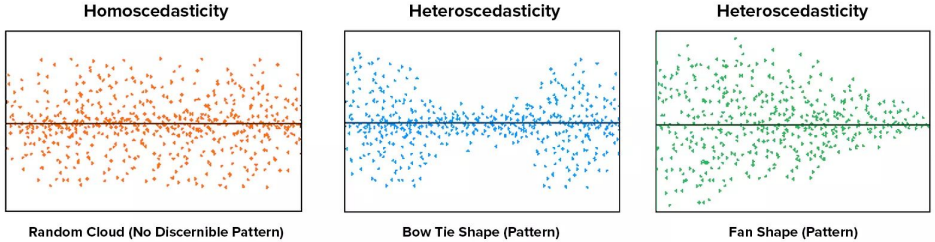


Age and loan amount appear to have no multicollinearity. We can include both in our model.

OLS Regression Task

Is our f(x) correct for this problem?

Do we have homoscedasticity?



Good

Not good

Not good

Data must be homoscedastic, meaning the error rate is evenly distributed at all outcome variable values.

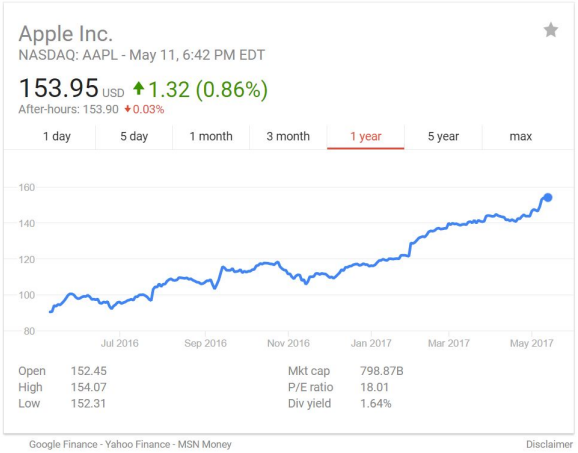
Error term or "noise" is the same across all values of the outcome variables. If homoscedasticity does not hold, cases with a greater error term will have outsized influence on the regression.

Source: Homoscedasticity

OLS Regression Task

Is our f(x) correct for this problem?

Do we have autocorrelation?



Autocorrelation is correlation between values of a variable and its delayed copy. For example, a stock's price today is correlated with yesterday's price

Autocorrelation commonly occurs when you work with time series.

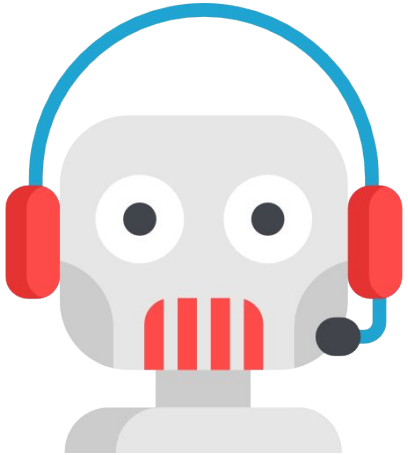
OLS Regression Task

Is our f(x) correct for this problem?

In the coding lab, we will go over code that will help you determine whether a linear model provides the best f(x)

OLS Assumptions

- ✓ Linear relationship between x and Y
- ✓ Normal distribution of variables
- ✓ No autocorrelation (Independent variables)
- ✓ Homoscedasticity
- ✓ Rule of thumb: at least 20 observations per independent variable in the analysis



We have signed off on all of our assumptions, which means we can confidently choose a linear OLS model for this task.

Let's start building our model!

OLS Regression Task

Is our f(x) correct for this problem?

Question! What happens if you don't have all the assumptions?

OLS Assumptions

- ✓ Linear relationship between x and Y
- ✓ Normal distribution of variables
- ✓ No autocorrelation (Independent variables)
- ✓ Homoscedasticity
- ✓ Rule of thumb: at least 20 observations per independent variable in the analysis

If these assumptions do not hold true our trend line will not be accurate. We have a few options:

1) **Transform our data** to fulfill the assumptions

2) **Choose a different model** to capture the relationship between x and Y

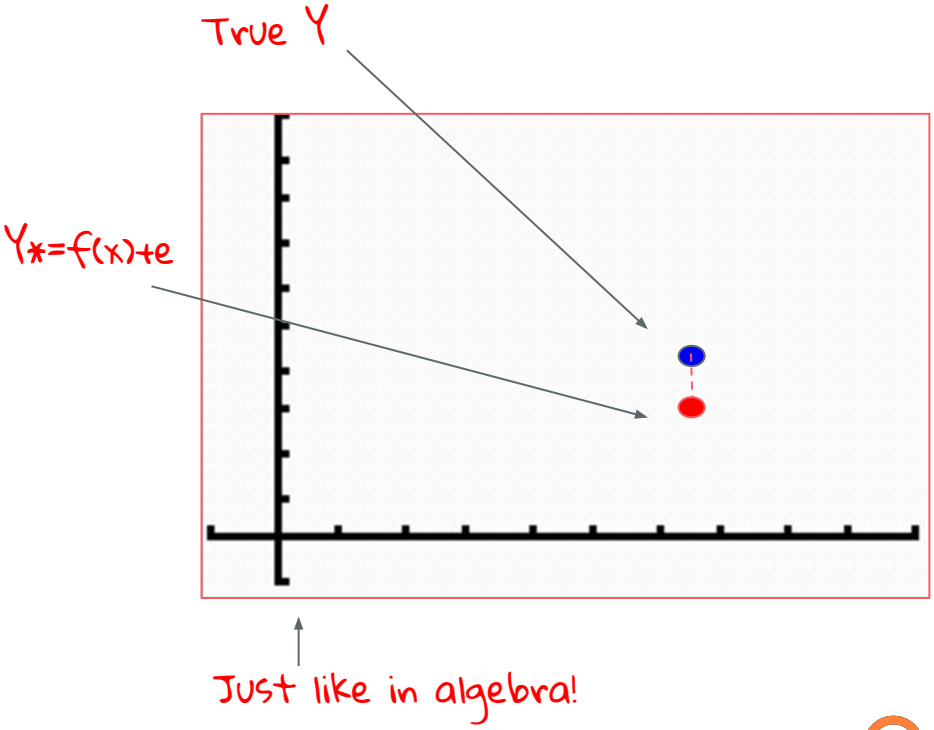
Yes! Linear regression is an appropriate model choice. Now what?

What is f(x) for a linear regression model?

Remember that all models involve a function f(x) that map an input x to a predicted Y (Y*). The goal of the function is to have a model that predicts Y* as close to true Y as possible.

f(x) for a linear model is:

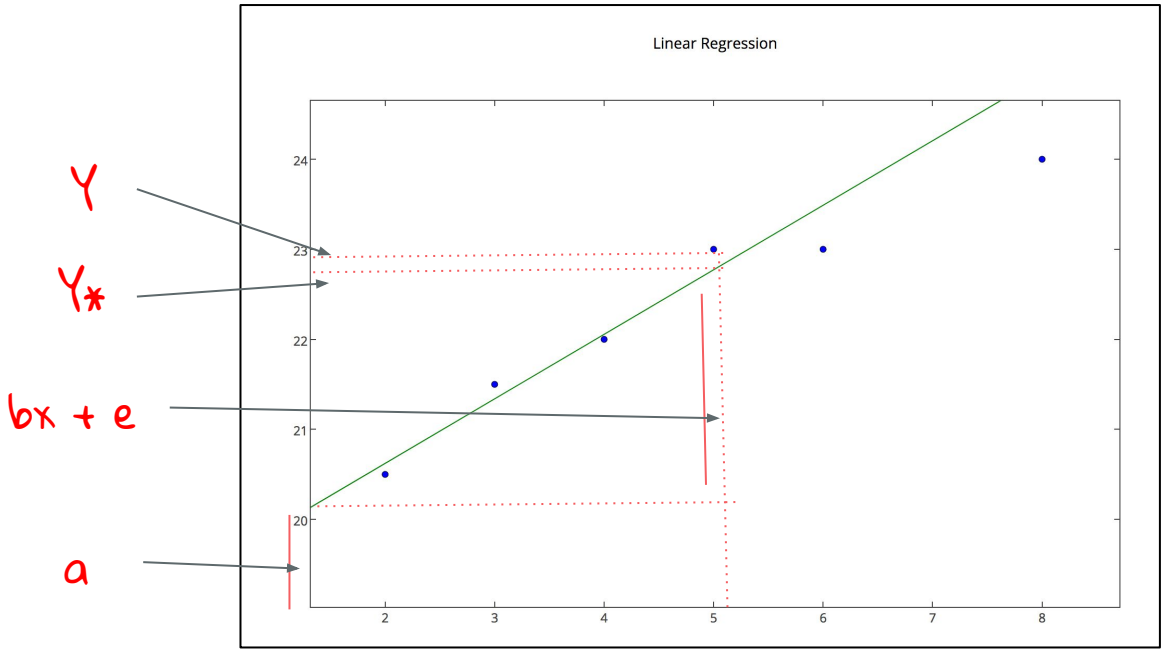
$$Y* = a + bx + e$$



What does $Y*=a + bx + e$ actually mean?

$$Y*=a + bx + e$$

parameters: a	The y-intercept of the regression line
parameters: b	The slope or gradient of the regression line. This determines how steep the line is and it's directionality
e	The irreducible error term, error our model cannot reduce
$Y*$	The predicted Y output of our function

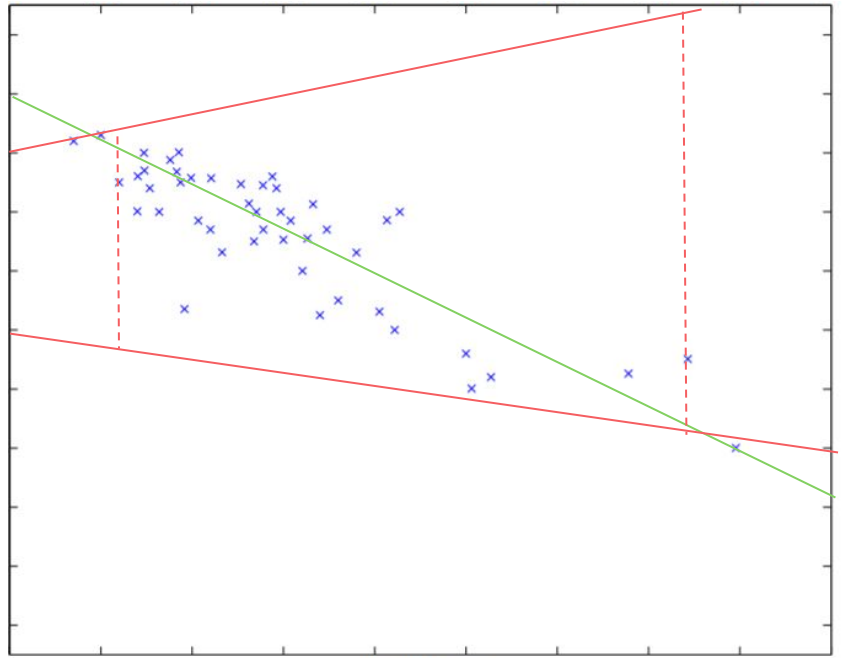


There are an infinite number of possible lines in a two dimensional space. How does our model choose the best one?

Linear regression is a learning algorithm. That is what makes it a machine learning model.

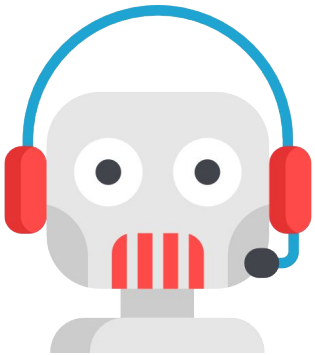
It learns to find the best trend line from an infinite number of possibilities. **How?**

First, we must understand what levers we control.



Parameters of a model are values that we control. The parameters in an OLS Model are a (intercept) and b (slope)

$Y* = a + bx + e$



In each model there are some things we cannot control, like e (irreducible error).

A model may also have **hyperparameters** which are set beforehand and not trained using data (no need to think too much about this now).

Parameters → Values that control the behavior of the model and are learnt through experience.

Hyperparameters → Higher level settings of a model that are fixed before training begins.

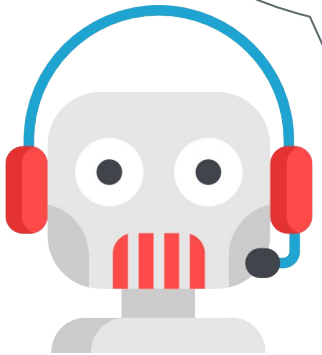
Our model can move a & b but not e.

Learning Methodology

How does our OLS model learn?

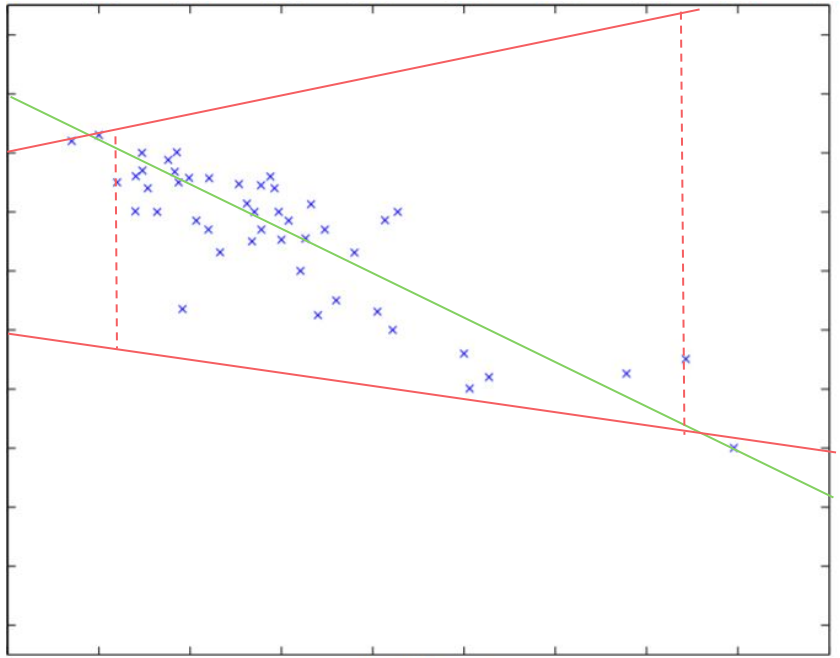
a and b are the only two levers our simple OLS model can change to get Y* closer to Y.

$Y* = a + bx + e$



How do I decide in what direction to change a and b?

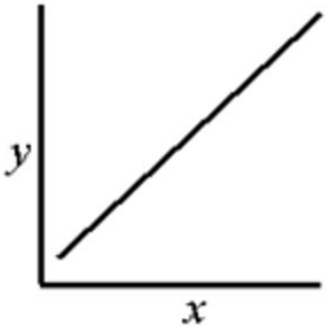
changing a shifts our line up or down the y intercept, +/- b changes the steepness of our line and direction



We can change a and b to move our line in space. Below is some intuition about how changing a and b affects f(x).

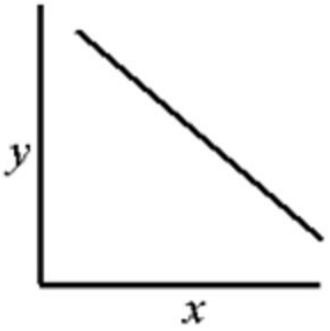
$$Y* = a + bx + e$$

a and b are our two parameters. changing a shifts our line up or down the y intercept. Negative a moves the y-intercept below 0.



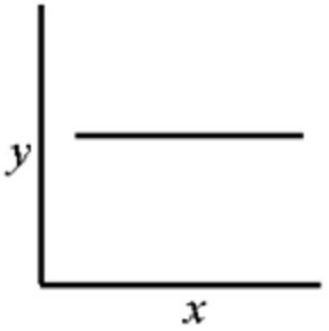
Positive slope

+ b means an upwards sloping line



Negative slope

- b means a downward sloping line



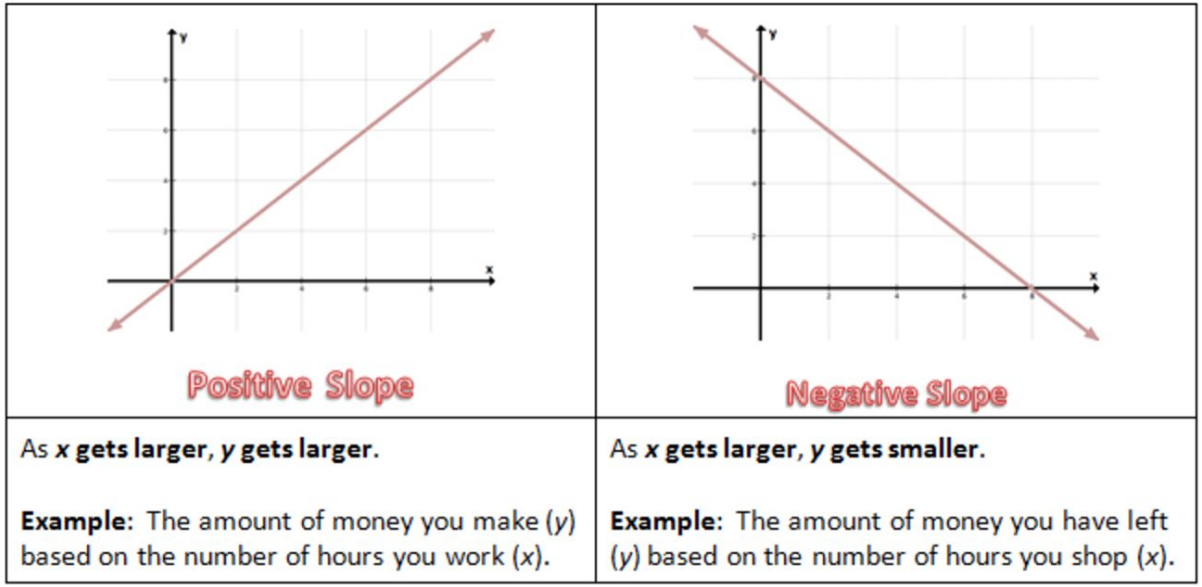
Zero slope

b=0 means there is no relationship between x and Y

Source: Intro to Stat - Introduction to Linear Regression

The directionality of b is very important. It tells us if Y gets smaller or larger when we increase x.

$Y* = a + bx + e$



+ b is a positive slope

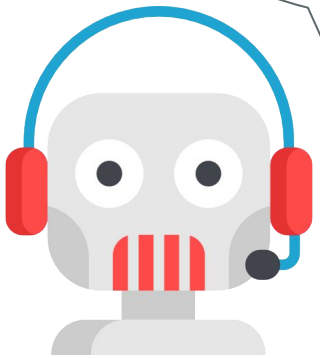
- b is a negative slope

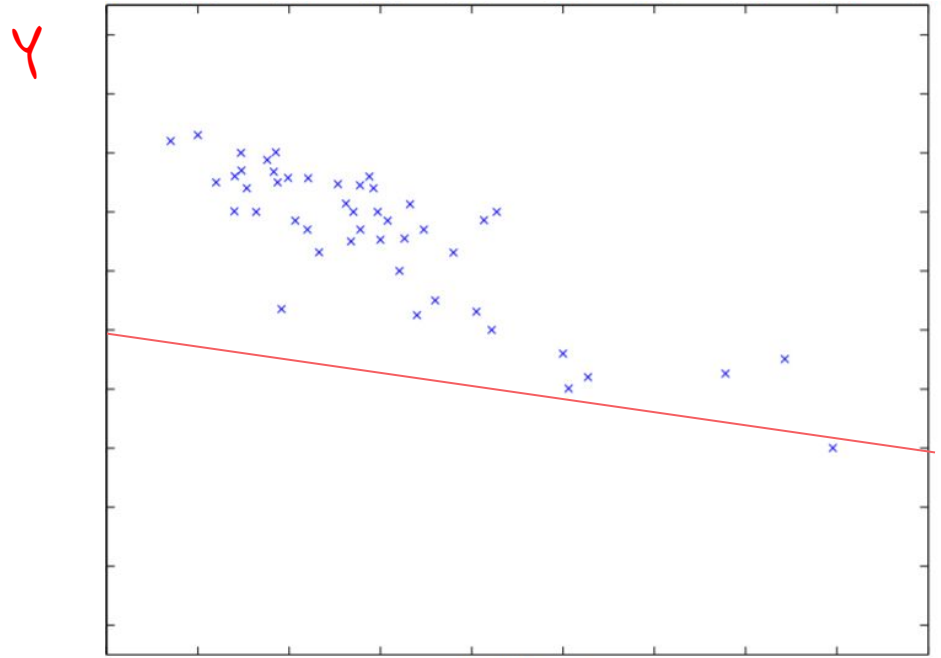
Source: Beginning Algebra - Coordinate System and graphing lines

Test your intuition.

$Y* = a + bx + e$

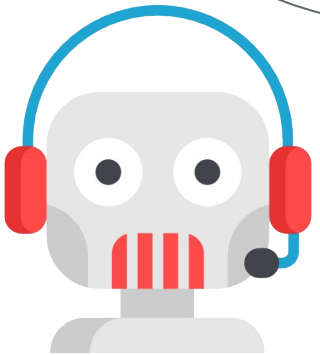
What happens if I increase a by 2?

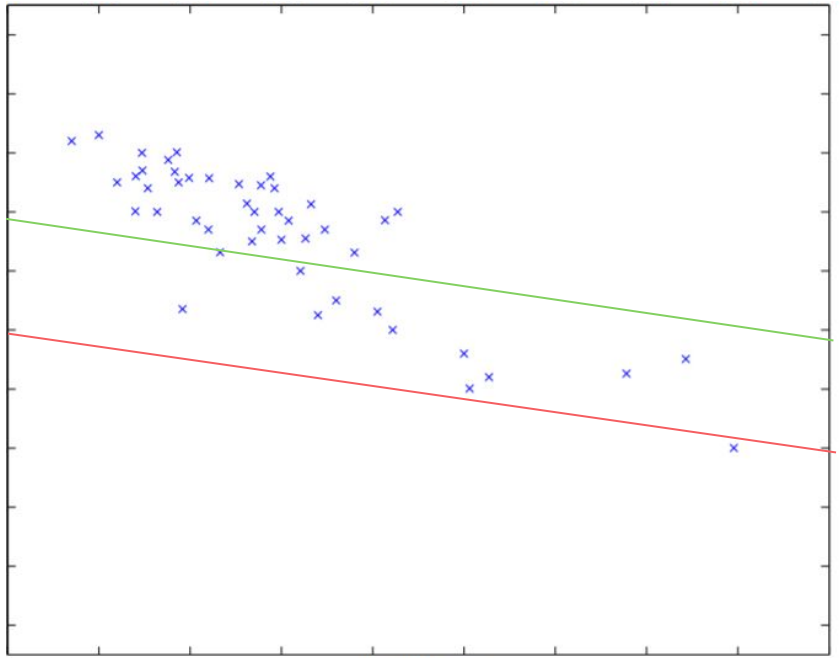




$Y* = a + bx + e$

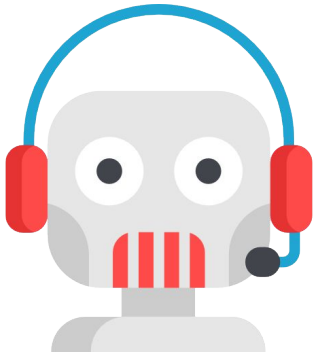
What happens if I increase a by 2?

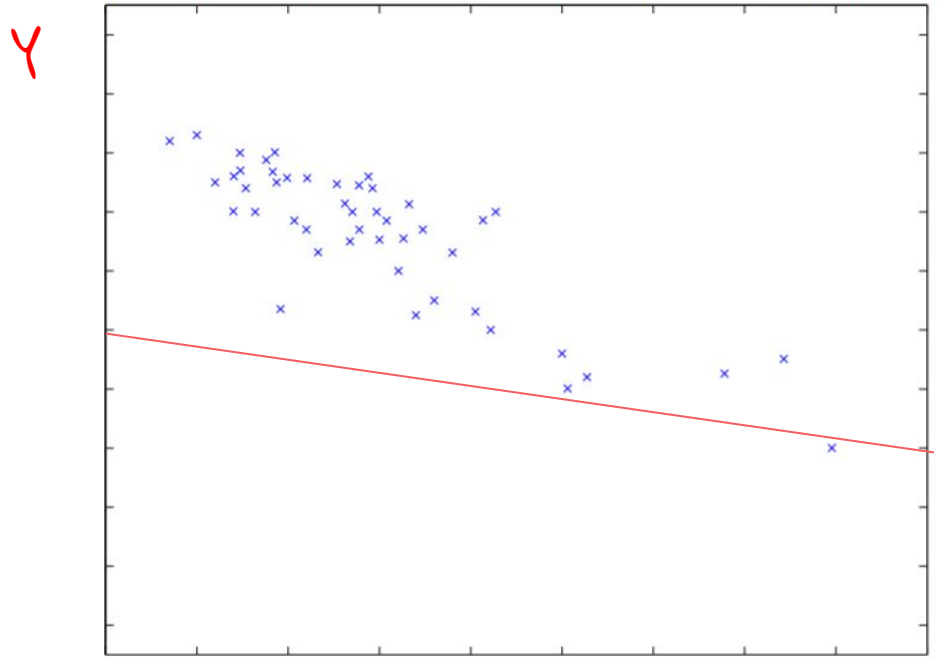




$Y* = a + bx + e$

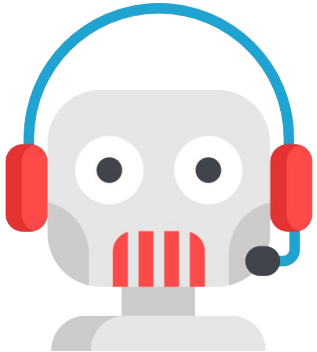
What happens if I increase b by 3?

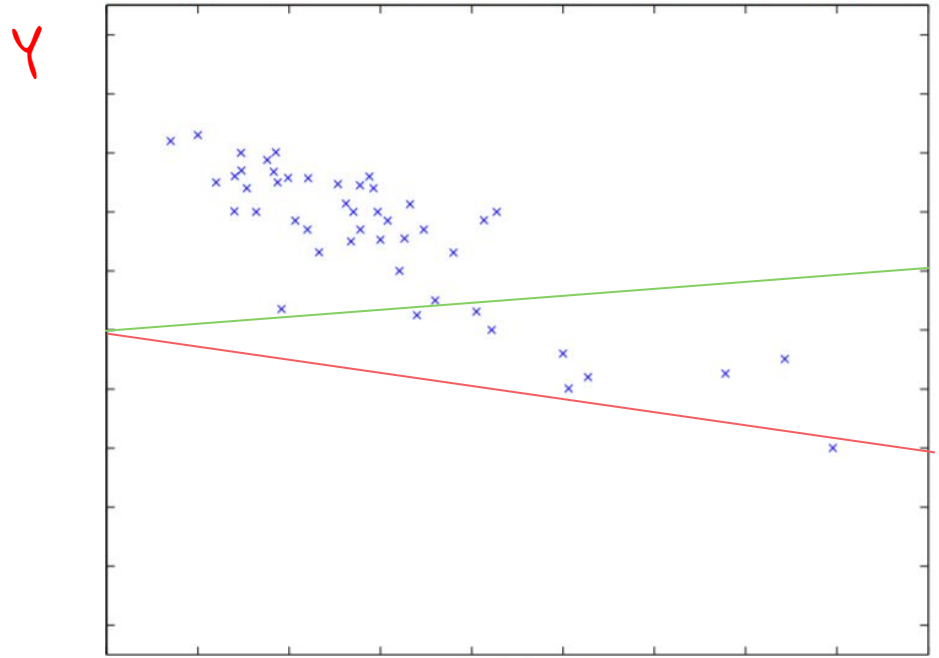




$Y* = a + bx + e$

What happens if I increase b by 3?

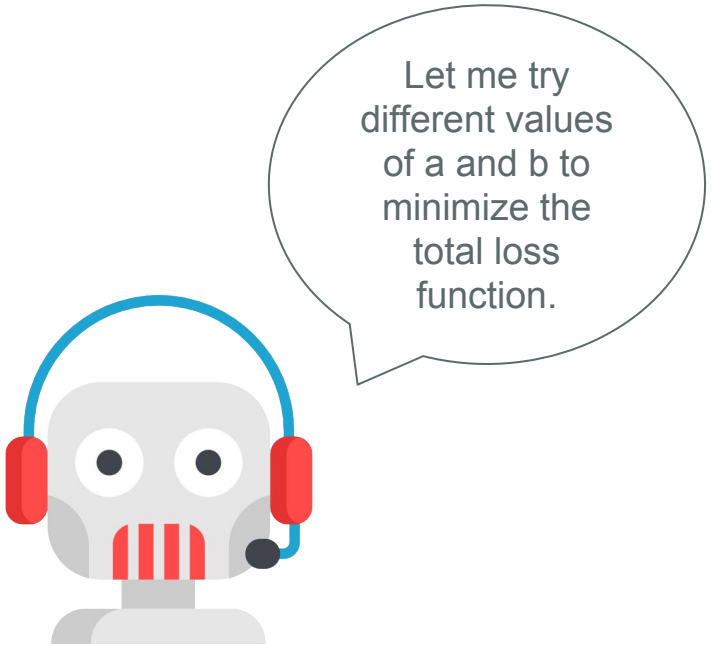




What parameters give us the best trend line?

We make a decision about how to change our parameters based upon our loss function.

$Y^* = a + bx + e$



Recall: **Our model starts with a random a and b**, and our job is to change a and b in a way that moves Y^* closer to the true Y.

You are in fact trying to reduce the distance between Y^* and true Y. We measure the distance using mean squared error. This is our loss function.

All supervised models have a loss function (sometimes also known as the cost function) they must optimize by changing the model parameters.

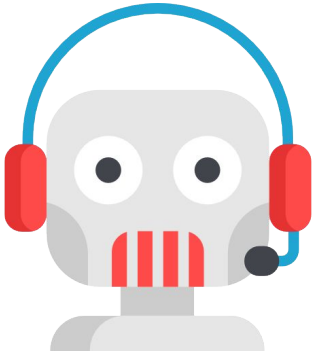
Learning Methodology

What is our loss function?

For every a and b we try, we measure the mean squared error. Remember MSE?

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

Mean squared error is a measure of how close Y* is to Y.



This job isn't done until I reduce MSE.

There are four steps to MSE:

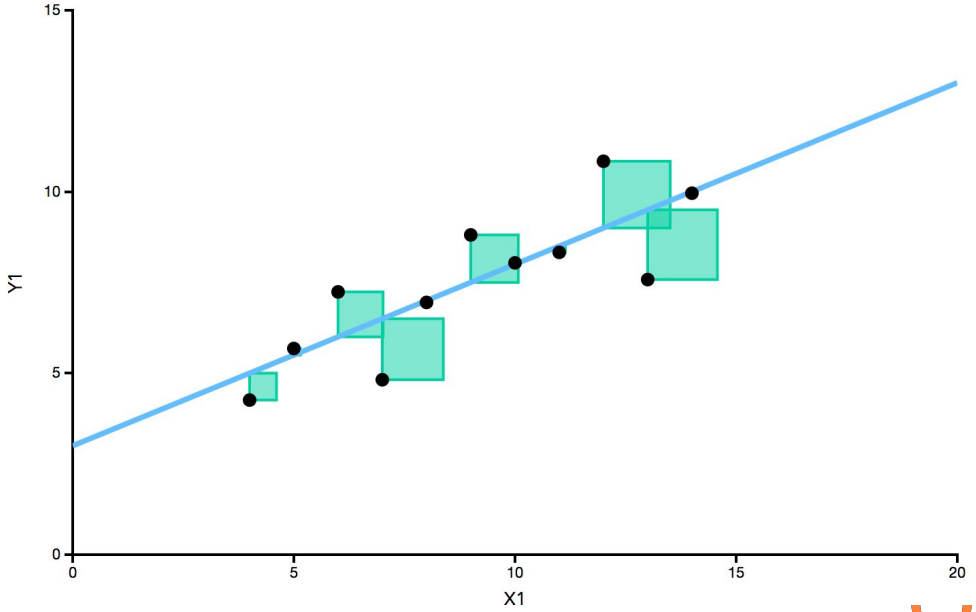
Y-Y*	For every point in our dataset, measure the difference between true Y and predicted Y.
^2	Square each Y-Y* to get the absolute distance, so positive values don't cancel out negative ones when we sum.
Sum	Sum across all observations so we get the total error.
mean $\sum_{i}^{n}$	Divide the sum by the number of observations we have.

Source: Intro to Stat -Introduction to Linear Regression

The green boxes in the chart below is the MSE for each data point. We sum and then take the mean across all data points to get the MSE.

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

Y-Y*	For every point in our dataset, measure the difference between true Y and predicted Y.
^2	Square each Y-Y* to get the absolute distance, so positive values don't cancel out negative ones when we sum.
Sum	Sum across all observations so we get the total error.
mean	Divide the sum by the number of observations we have.



Source: Seeing Theory - Regression

Learning Methodology

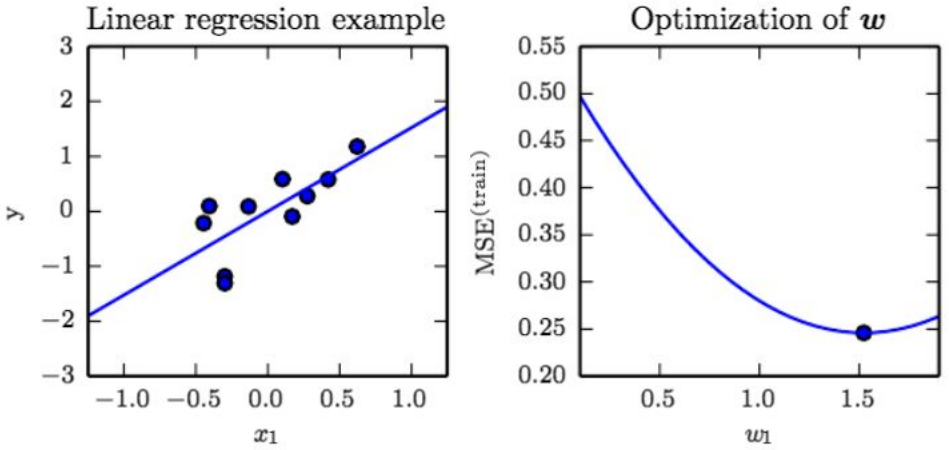
Optimization Process

The process of changing a and b to reduce MSE is called **learning**. It is what makes OLS regression a machine learning algorithm.

For every combination of a and b we choose there is an associated MSE. The learning process involves updating a and b in order to reach the global minimum.

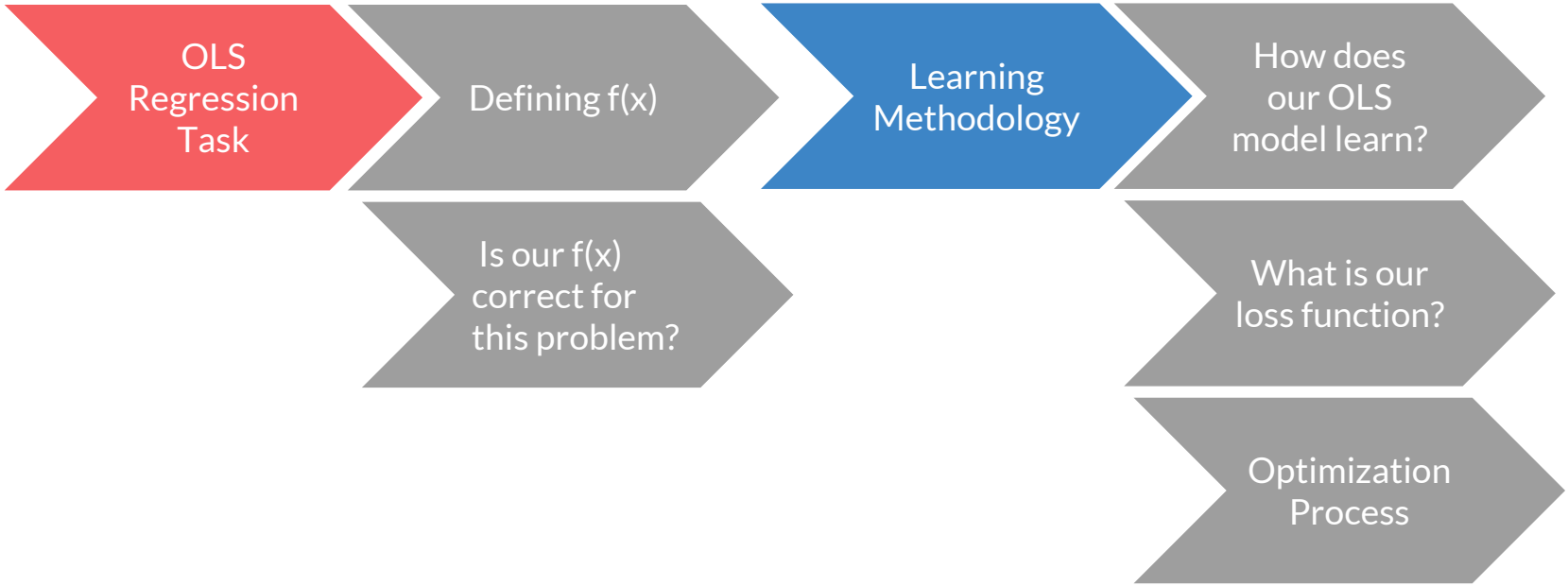
The learning process for OLS is technically called **learning by gradient descent**.

Optimization of MSE:



Source: Seeing Theory - Regression

Next, let's look at model validation.



Model Validation

Thoroughly validating your model is crucial.

Validation is a process of evaluating your model performance.

We have to evaluate a model on two critical aspects:

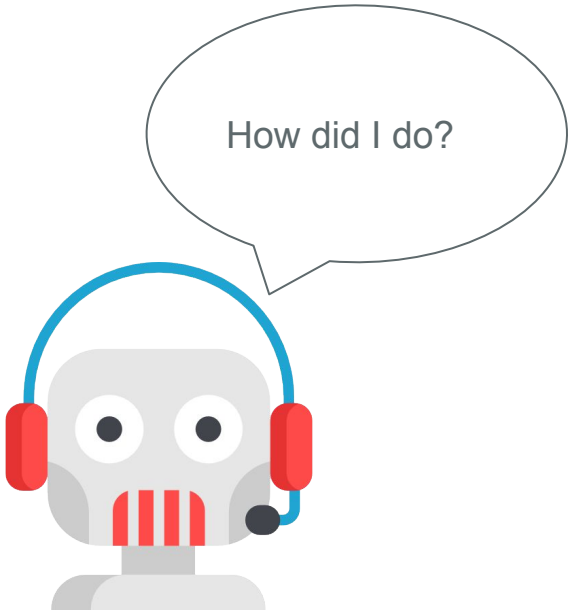
1. **How close the model's Y* is to true Y**
2. **How well the model performs in the real world (i.e., on *unseen* data).**

Common validation metrics:

1. R2
2. Adjusted R2
3. MSE (a loss function *and* a measure of performance)

Let's take a look at these metrics...

For an OLS regression, we will introduce three measures of model performance: R2, Adjusted R2 and MSE.



Mean Squared Error

We have already introduced MSE. This serves as both a loss function and an evaluation of performance.

R^2

Explained Variation / Total Variation
Increases as number of x's increase.

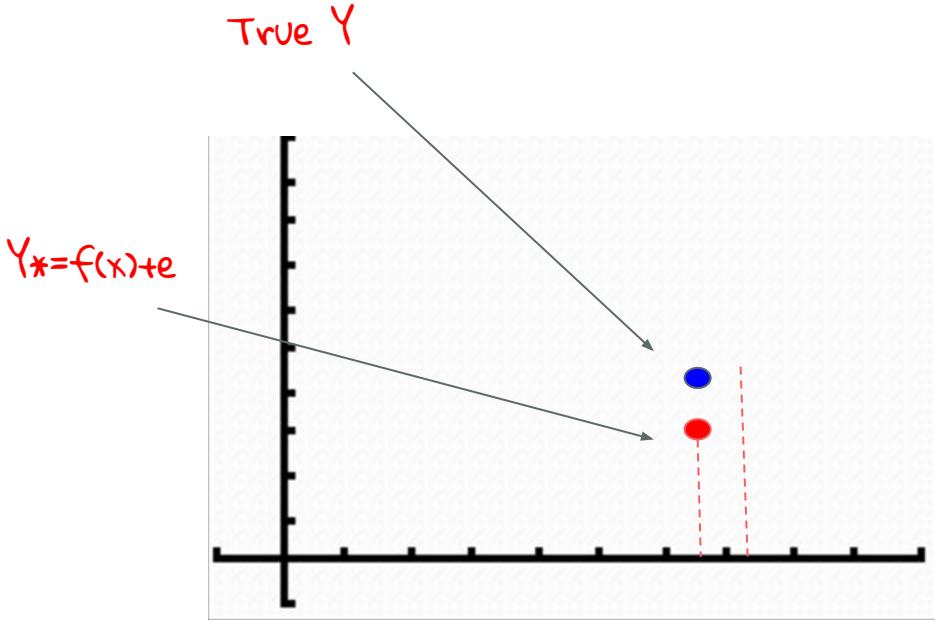
Adjusted R^2

Explained Variation / Total Variation, adjusted for the number of features present in the model

R2 and Adjusted R2 answer the question, *how much of y's variation can be explained by our model?*

This reminds us of MSE, but **R2 metrics are scaled to be between 0 and 1.**

Note: Adjusted R^2 is preferable to R^2, because R^2 increases as you include more features in your model. This may artificially inflate R^2.



If Y* is explained variation, what's left between Y* and Y is unexplained variation

In a regression output, you can find R-squared metrics here:

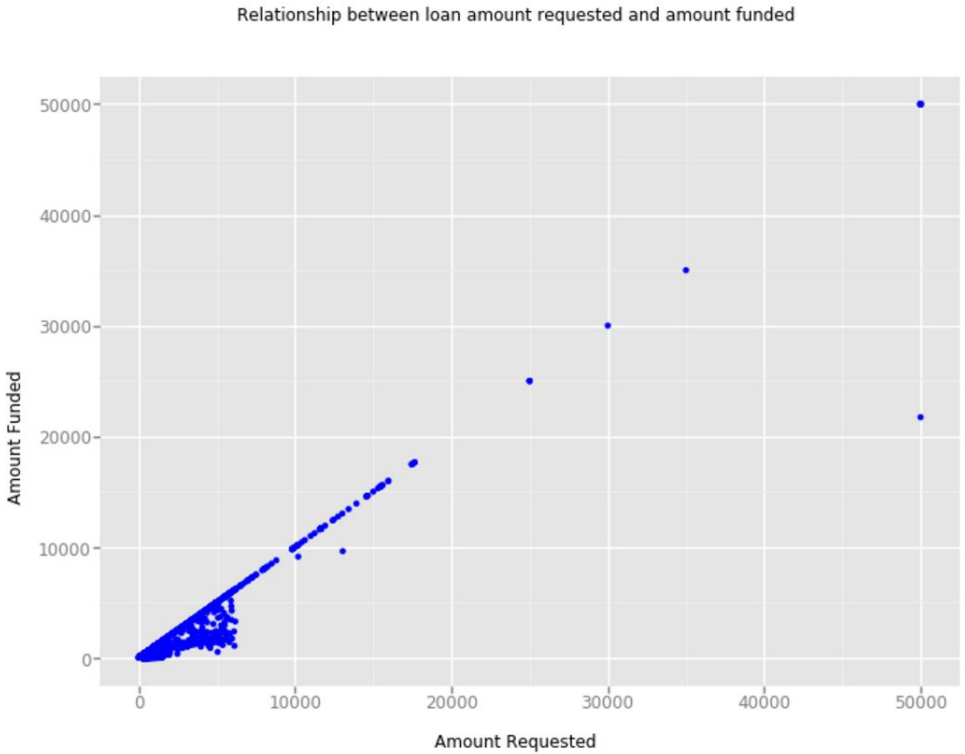
OLS Regression Results

Dep. Variable:	log_loan_amount	R-squared:	0.356
Model:	OLS	Adj. R-squared:	0.355
Method:	Least Squares	F-statistic:	1836.
Date:	Sun, 28 May 2017	Prob (F-statistic):	0.00
Time:	15:06:29	Log-Likelihood:	-73913.
No. Observations:	89811	AIC:	1.479e+05
Df Residuals:	89783	BIC:	1.481e+05
Df Model:	27		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
Intercept	42.3433	1.330	31.840	0.000	39.737	44.950
sector[T.Arts]	-0.0805	0.034	-2.376	0.017	-0.147	-0.014
sector[T.Clothing]	0.0793	0.009	9.215	0.000	0.062	0.096
sector[T.Construction]	0.0292	0.017	1.736	0.083	-0.004	0.062
sector[T.Education]	-0.1096	0.015	-7.312	0.000	-0.139	-0.080

Source: This is a snippet of the output from Notebook 2.

R2 can give us causation where correlation couldn't!



R-squared reminds us of correlation, but there is an important difference.

Correlation measures the **association** between x and y.

R-squared measures how much **y is explained by x.**

Now we know how well the model performs on the data we have, how do we predict how it will do in the real world?

We need a way to quantify the way our model performs on unseen data. In an ideal world, we would go out and find **new data** to test our model on.

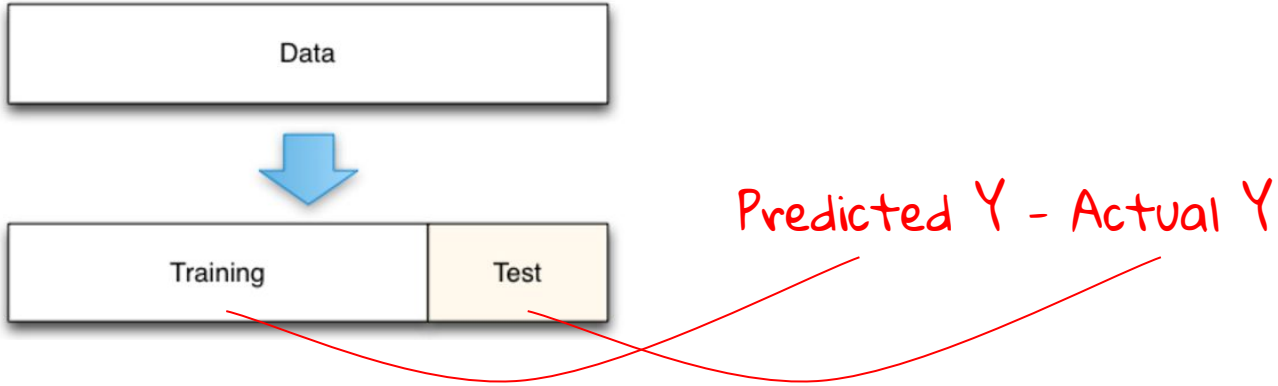
However, this is often not realistic as we are constrained by time and resources. Instead of doing this, we can instead **split** our data into two portions: **training and test data**.

We will use a portion of our data to train our model, and the rest of our data (which is "unseen" by the model, like real world data) to test our model.

Performance

Ability to generalize to unseen data

We split our labelled data into training and test. The test represents our **unseen future data**.



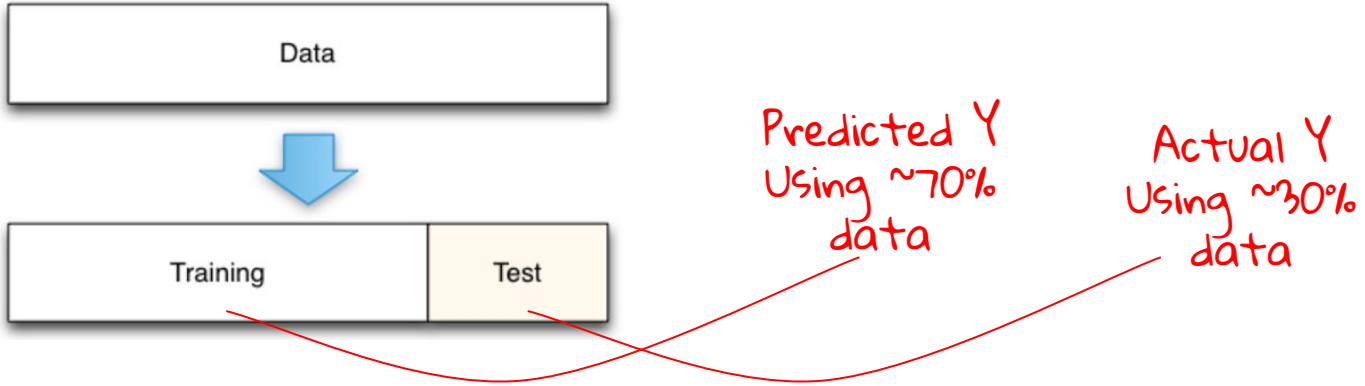
- Randomly split data into “training” and “test” sets
- Use regression results from “training” set to predict“test” set
- Compare Predicted Y to Actual Y

Source: Fortmann-Roe, Accurately Measuring Model Prediction Error, http://scott.fortmann-roe.com/docs/MeasuringError.html

Performance

Ability to generalize to unseen data

We split our labelled data into training and test. The test represents our **unseen future data.**



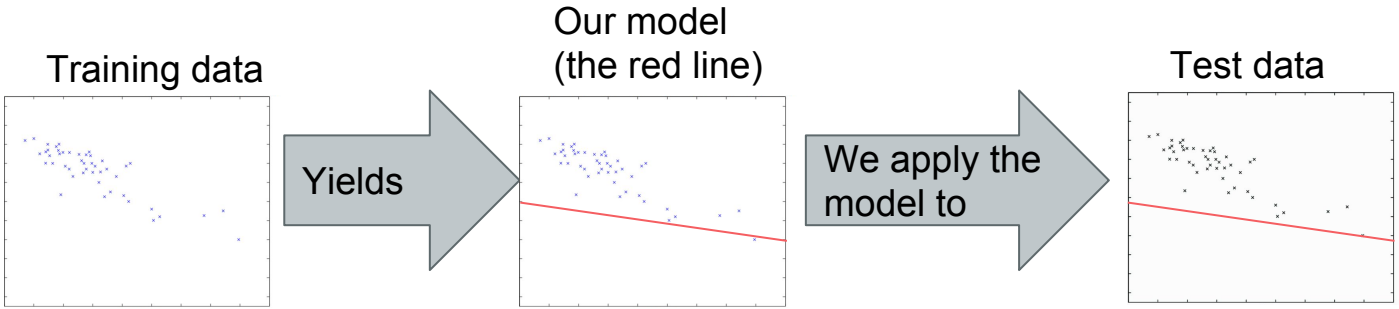
The test data is "unseen" in that the algorithm doesn't use it to train the model!

Performance

Ability to generalize to unseen data

Don't confuse our use of loss functions with our use of test data!

It is important to clarify here that we are using train Y*- train Y to train the model, which **is different than** the test Y*- test Y we use to evaluate how well the model can generalize to unseen data.



We use train Y*- train Y in our loss functions that train the model.

We use test Y* - test Y to see how well our model can be generalized to unseen data, or data that wasn't used to train our model..

Splitting train and test data is **extremely important!**

Evaluating whether or not a model can be generalized to unseen data is very important - in fact, being able to predict it is often the whole point of creating a model in the first place.

If we do not evaluate how well a model can generalize to outside data, **there is a danger that we are creating a model that is too specific to our dataset** - that is, a model that is GREAT at predicting this particular dataset, but is USELESS at predicting the real world.

What does this look like?

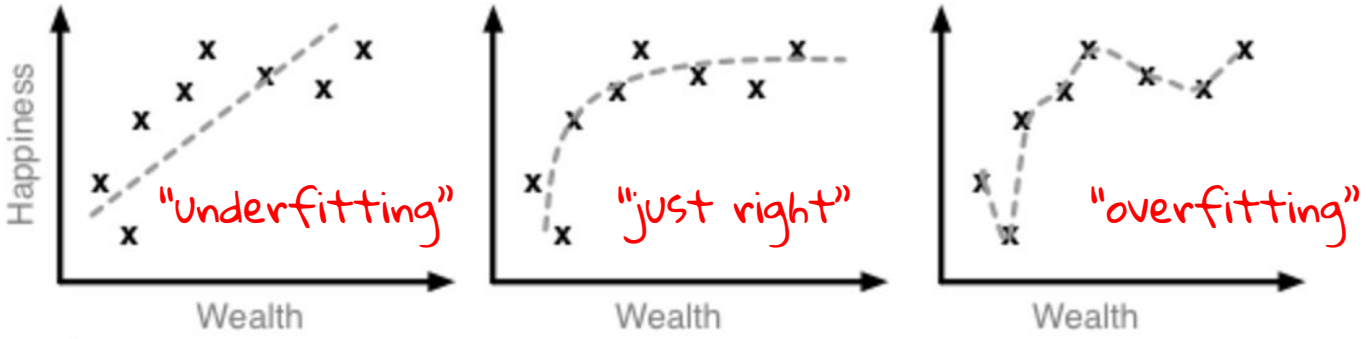
Performance

Ability to generalize to unseen data

Splitting train and test data is **extremely important!**

Here, we are using wealth to predict happiness. The dotted lines are the models generated by ML algorithms.

- On the left, the model is not useful because it is **too general** and does not capture the relationship accurately.
- On the right, the model is not useful because **not general enough** and captures every single idiosyncrasy in the dataset. This means the model is too specific to this dataset, and cannot be generalized to different datasets.



We want to have a model that is just right - it is accurate enough within the dataset, and is general enough to be applied outside of the dataset!

Performance

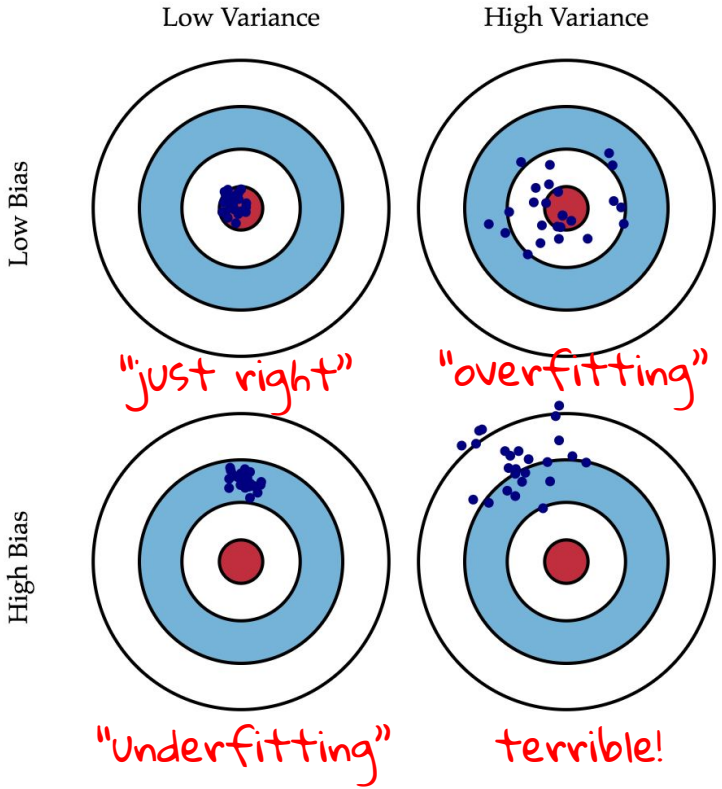
Ability to generalize to unseen data

This concept of a model being "just right" is also called the **bias-variance trade-off.**

Bias is how accurate the model is at predicting the dataset.

Variance is how sensitive the model is to small fluctuations in the training set.

Ideally, we would have both low bias and low variance.



Source: Fortmann-Roe, Accurately Measuring Model Prediction Error.
http://scott.fortmann-roe.com/docs/MeasuringError.html

Don’t worry about the specifics of the bias-variance trade-off for now - we will return to this concept regularly throughout the course and in-depth in the next module.

Let’s turn to another aspect of performance that is very important: **feature performance.**

Performance

Feature Importance

The performance of model components is just as important as the performance of the model itself

Understanding **feature importance** allows us to:

1) Quantify which feature is driving explanatory power in the model
2) Compare one feature to another
3) Guide final feature selection

Evaluating the performance of features becomes important when we start using more sophisticated linear models where f(x) includes more than one explanatory variable. Let's introduce that concept and then return here.

OLS Task

Univariate vs. Multivariate as f(x)

Deciding whether a univariate or multivariate regression is the best model for your problem is part of the task.

Univariate

One explanatory variable

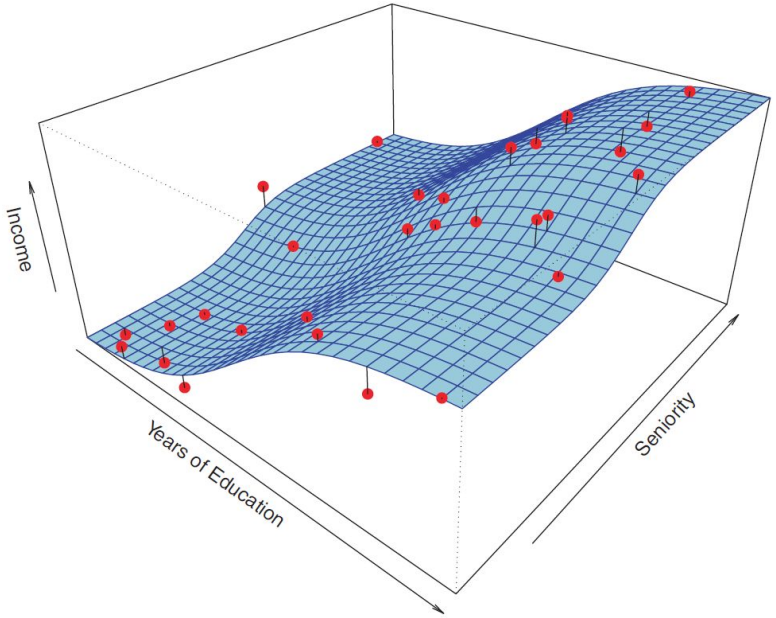
E.g. Trying to predict malaria using only patient's temperature

Multivariate

Multiple explanatory variables.

E.g. Trying to predict malaria using patient's temperature, travel history, whether or not they have chills, nausea, headache.

Many of the relationships we try and model are more complicated than a univariate regression. Instead, we use a multivariate regression.

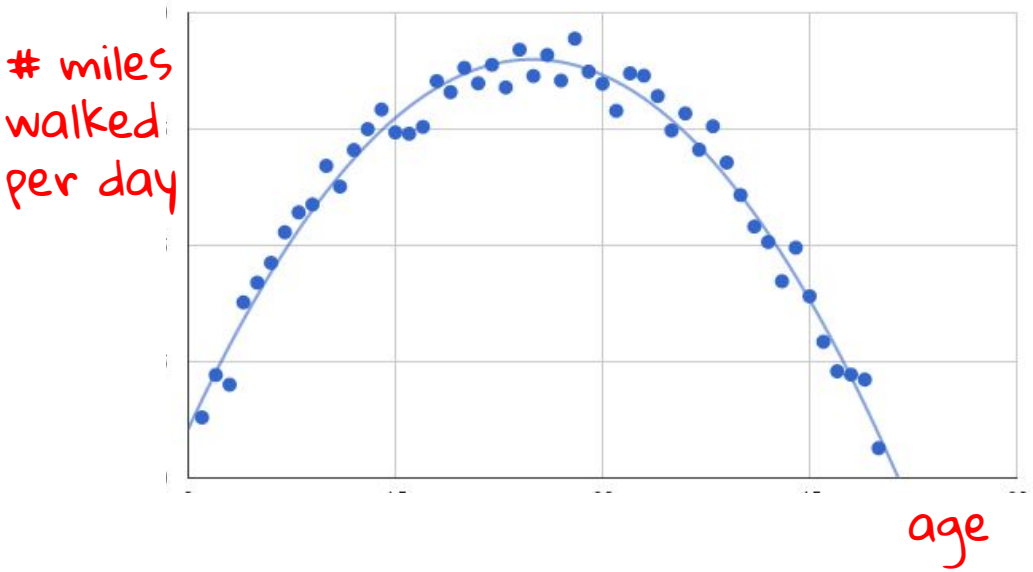


$$\text{Income} = a + b_1(\text{Seniority}) + b_2(\text{Years of Education}) + e$$

This is an example of linear regression with 2 explanatory variables in 3 dimensions. **Extend this to n variables in n+1 dimensions.**

Source: James, Witten, Hastie and Tibshirani. Introduction to Statistical Learning with Applications in R.

Regressions can also be non-linear!



Miles walked per day =
$a + b_1(Age)^2 + b_2(Age) + e$

In this example, we see that the relationship between your mobility and your age increases, then decreases after some point. This is an **example of a non-linear regression**, which is best explained by a quadratic equation.

Source: [link]

When we model using a multivariate regression, feature selection becomes an important step. What explanatory variables should we include?

Feature selection is often the difference between a project that fails and succeeds. Some common ways of doing feature selection:

Qualitative Research

Literature review of past work done

Using Other Models

Quantify feature importance (we'll see this later with decision trees)

Exploratory Analysis

Sanity checks on data; human intuition of what would influence outcome

Analyzing Linear Regression Output

Look at each feature's coefficient and p-value

Performance

Feature Importance

How do we assess feature importance in a linear regression?

OLS Regression Results

Dep. Variable:	log_loan_amount	R-squared:	0.356
Model:	OLS	Adj. R-squared:	0.355
Method:	Least Squares	F-statistic:	1836.
Date:	Sun, 28 May 2017	Prob (F-statistic):	0.00
Time:	15:06:29	Log-Likelihood:	-73913.
No. Observations:			
Df Residuals:			
Df Model:			
Covariance Type:	nonrobust		

Each feature has a **coefficient** and a **p-value**.

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
Intercept	42.3433	1.330	31.840	0.000	39.737	44.950
sector[T.Arts]	-0.0805	0.034	-2.376	0.017	-0.147	-0.014
sector[T.Clothing]	0.0793	0.009	9.215	0.000	0.062	0.096
sector[T.Construction]	0.0292	0.017	1.736	0.083	-0.004	0.062
sector[T.Education]	-0.1096	0.015	-7.312	0.000	-0.139	-0.080

Performance

Feature Importance

How do we assess feature importance in a linear regression?

Each feature has a:

1. **Coefficient**
2. **P-value**

The output of a linear regression is a model:

$$Y* = intercept + coef*feature$$

The size of the coefficient = **amount of influence that the feature has on y.** Whether or not the feature is negative or positive is the **direction of the relationship that the feature has with y**.

Performance

Feature Importance

How do we assess feature importance in a linear regression?

Each feature has a:

1. **Coefficient**
2. **P-value**

Expressed as a %

A huge coefficient is great, but how much **confidence** we have in that coefficient **is dependent on that coefficient's p-value.**

In technical terms, the p-value is the probability of getting results as extreme as the ones observed, if the coefficient were actually zero. It answers the question, "**Could I have gotten my result by random chance?**"

A small p-value (<= 0.05, or 5%) says that the result is probably not random chance - great news for our model!

Heads or tails?
It's random!

Performance

Feature Importance

How would you assess this feature using p-value and size of the coefficient?

OLS Regression Results

Dep. Variable:	log_loan_amount	R-squared:	0.356
Model:	OLS	Adj. R-squared:	0.355
Method:	Least Squares	F-statistic:	1836.
Date:	Sun, 28 May 2017	Prob (F-statistic):	0.00
Time:	15:06:29	Log-Likelihood:	-73913.
No. Observations:	89811	AIC:	1.479e+05
Df Residuals:	89783	BIC:	1.481e+05
Df Model:	27		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
Intercept	42.3433	1.330	31.840	0.000	39.737	44.950
sector[T.Arts]	-0.0805	0.034	-2.376	0.017	-0.147	-0.014
sector[T.Clothing]	0.0793	0.009	9.215	0.000	0.062	0.096
sector[T.Construction]	0.0292	0.017	1.736	0.083	-0.004	0.062
sector[T.Education]	-0.1096	0.015	-7.312	0.000	-0.139	-0.080

Performance

Feature Importance

How would **you** assess this feature using p-value and size of the coefficient?

OLS Regression Results

Dep. Variable:	log_loan_amount	R-squared:	0.356
Model:	OLS	Adj. R-squared:	0.355
Method:	Least Squares	F-statistic:	1836.
Date:	Sun, 28 May 2017		
Time:	15:06:29		
No. Observations:	89811		
Df Residuals:	89783		
Df Model:	27		
Covariance Type:	nonrobust		

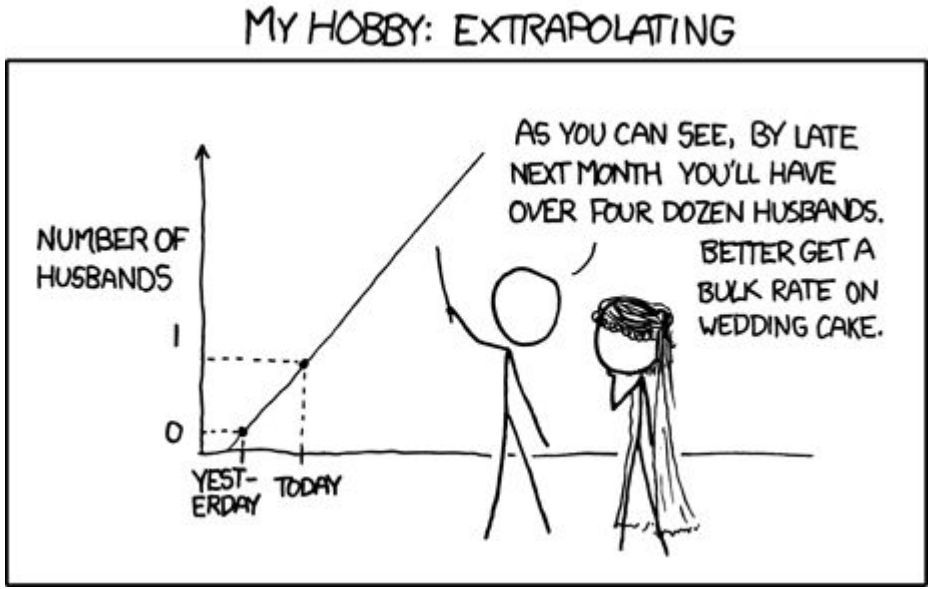
A person in the clothing sector will, on average, get a higher loan amount, but only by very little. I'm reasonably confident in this conclusion.

	coef	std err	t	P>|t|	[95.0% Conf.	Int.]
Intercept	42.3433	1.330	31.840	0.000	39.737	44.950
sector[T.Arts]	-0.0805	0.034	-2.376	0.017	-0.147	-0.014
sector[T.Clothing]	0.0793	0.009	9.215	0.000	0.062	0.096
sector[T.Construction]	0.0292	0.017	1.736	0.083	-0.004	0.062
sector[T.Education]	-0.1096	0.015	-7.312	0.000	-0.139	-0.080

A few last thoughts...

Extrapolation is the act of inferring unknown values based on known data.

Even validated algorithms are subject to irresponsible extrapolation!



Linear regression has almost countless potential applications, provided we interpret carefully.

Source: https://xkcd.com/605/

“All models are wrong, some are useful.”

- George E.P. Box,
British statistician

Module Checklist

- ✓ Linear regression
  - ✓ Relationship between two variables (x and y)
    - ✓ Formalizing f(x)
    - ✓ Correlation between two variables
    - ✓ Assumptions
  - ✓ Feature engineering and selection
  - ✓ Univariate regression, Multivariate regression
  - ✓ Measures of performance (R2, Adjusted R2, MSE)
  - ✓ Overfitting, Underfitting
  - ✓ Learning process: Loss function and Mean Squared Error

Advanced resources

Want to take this further? Here are some resources we recommend:

- Textbooks
  - An Introduction to Statistical Learning with Applications in R (James, Witten, Hastie and Tibshirani): Chapters 2.1, 3, 4, 6
  - The Elements of Statistical Learning: Data Mining, Inference, and Prediction (Hastie, Tibshirani, Friedman): Chapters 3, 4
- Online resources
  - Analytics Vidhya's guide to understanding regression: https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/
  - Brown University's introduction to probability and statistics, http://students.brown.edu/seeing-theory/
- If you are interested in more sophisticated regression models, search:
  - Logistic regression, Polynomial regression, Interactions
- If you are interested in additional ways to solve multicollinearity, search:
  - Eigenvectors, Principal Components Analysis

You are on fire! Go straight to the next module here.

Need to slow down and digest? Take a minute to write us an email about what you thought about the course. All feedback small or large welcome!

Email: sara@deltanalytics.org

Congrats! You finished module 3!

Find out more about Delta's machine learning for good mission here.