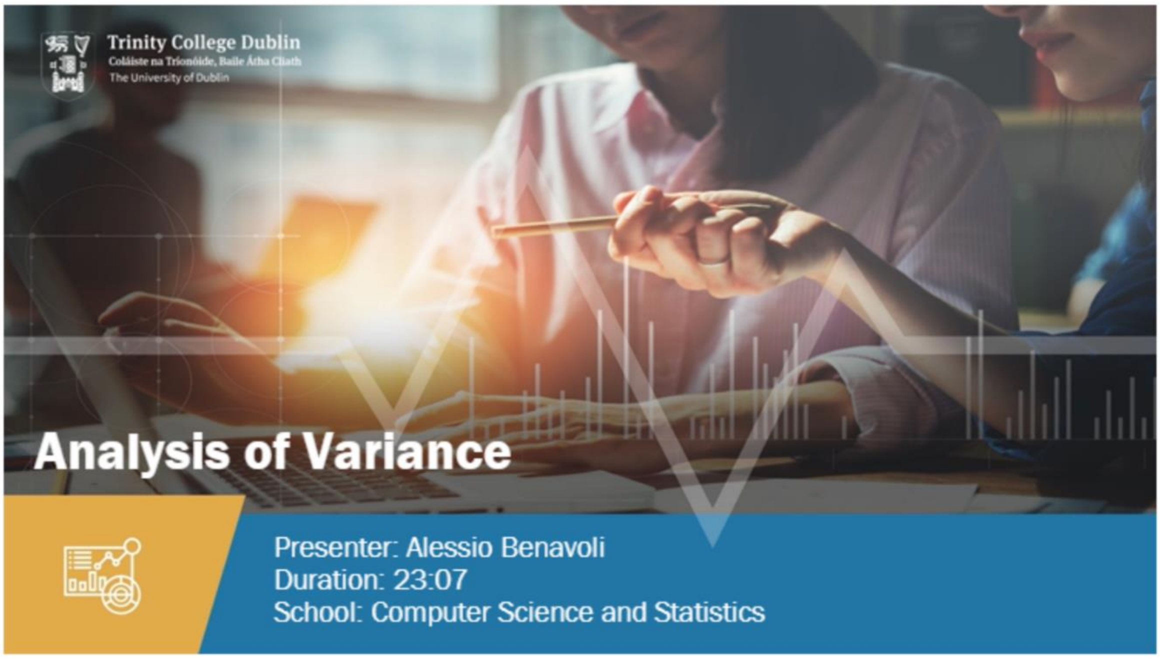
Analysis of Variance

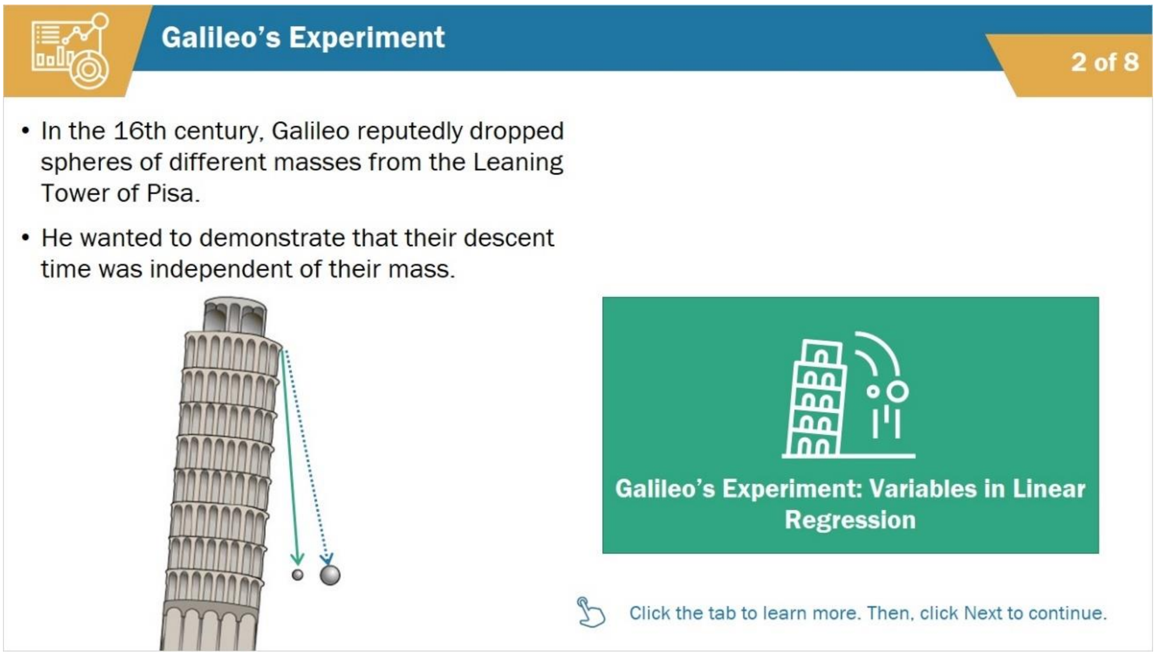
Contents

Slide 1: **Introduction**



My name is **Alessio Benavoli** and I am **the instructor for this session**. During this presentation I will discuss how we can include polynomial and interaction terms in linear regression. We will use a motivational example to introduce the topic: an experiment performed by Galileo on the leaning tower of Pisa. We will also discuss how we can deal with categorical variables in linear regression. We will see how to perform model selection using Anova and AIC criterion. We will then conclude the session with an application to the mcars dataset.

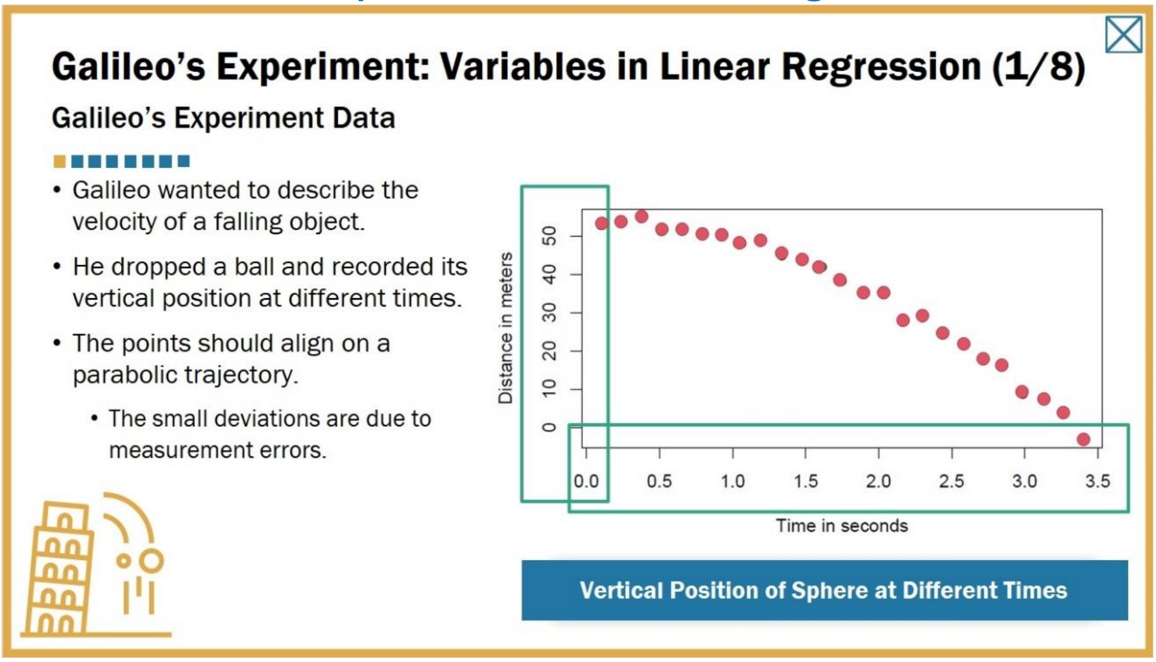
Slide 2: **Galileo's Experiment**



In the 16th century, Galileo is said to have dropped two spheres of different masses from the Leaning Tower of Pisa to demonstrate that their time of descent was independent of their mass.

Click the tab to learn more about dealing with variables in linear regression. When you are ready, click Next to continue.

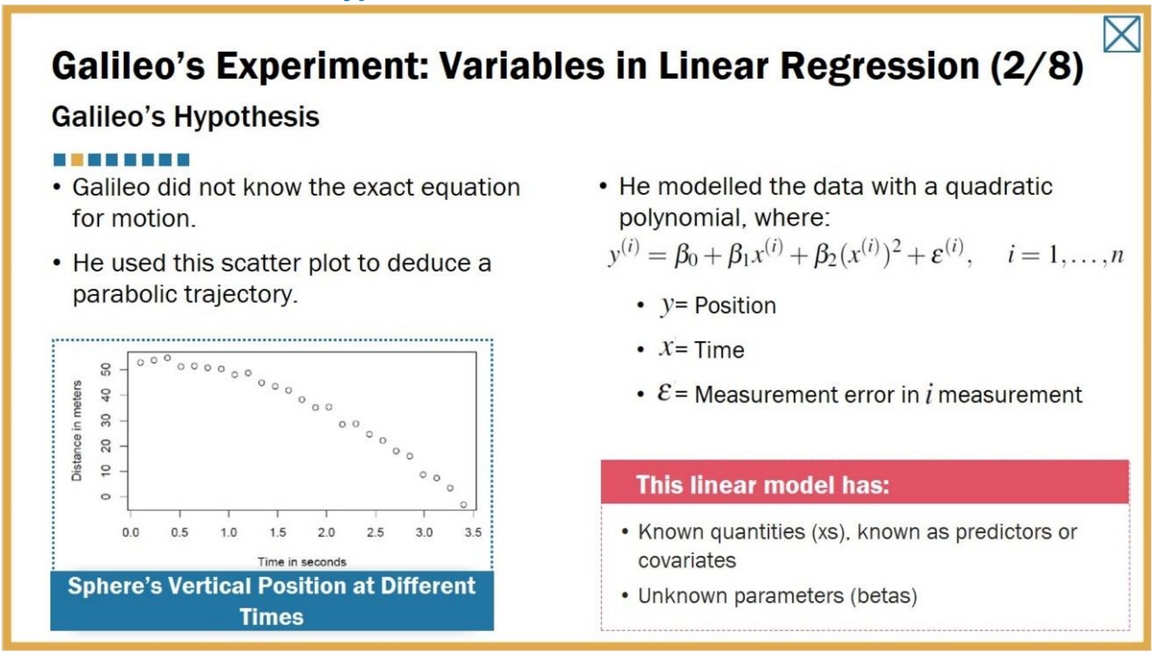
Tab 1.1: Galileo's Experiment: Variables in Linear Regression



Imagine you are Galileo trying to describe the velocity of a falling object. You climb the Tower of Pisa and drop a ball and also record its vertical position at different times. The scatter plot shows the vertical position y-axis, at different time instants x-axis.

The points should align on a parabolic trajectory: the small deviations are due to measurement errors (there weren't precise measurement instruments at Galileo's time).

Tab 1.2: Galileo's Hypothesis



Galileo didn't know the exact equation for the motion, but by looking at this plot deduced a parabolic trajectory. So, he modelled the data with the following quadratic polynomial:

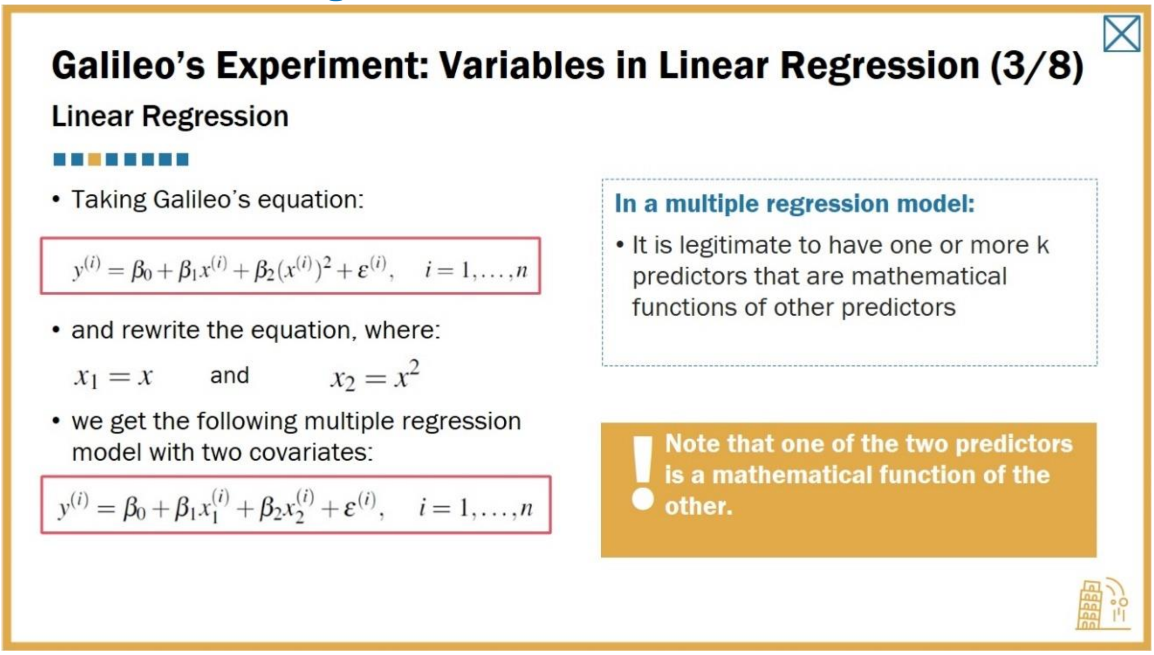
$$y^{(i)} = \beta_0 + \beta_1 x^{(i)} + \beta_2 (x^{(i)})^2 \ + \varepsilon^{(i)}, i = 1, \ldots, n$$

where the y represents the position, x represents the time, and epsilon accounts for measurement error in the i-th measurement.

This is a linear model, because it is a linear combination of known quantities (the xs) referred to as predictors or covariates through unknown parameters (the betas).

The model is linear with respect to the unknown betas.

Tab 1.3: Linear Regression

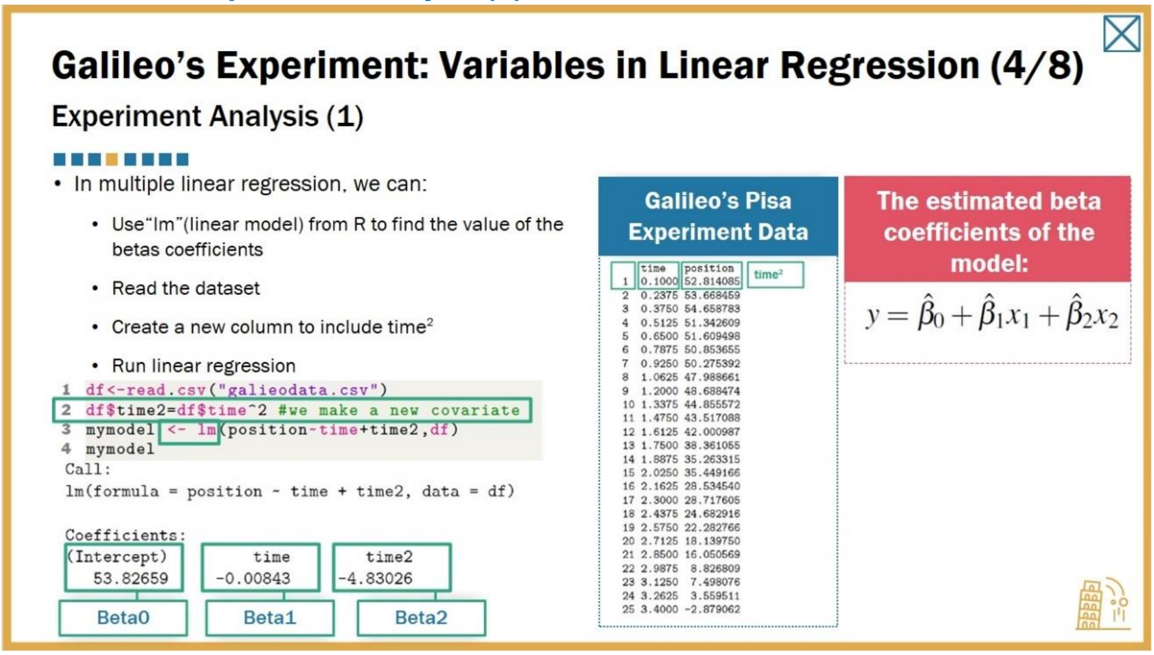


If we take Galileo's equation and rewrite it, denoting x by x1 and x-squared by x2, we obtain the model you see in the second equation.

We can now see that we are simply considering a multiple regression model with two covariates (or predictors).

Notice that one of the two predictors is a mathematical function of the other one: x2 is in fact equal to x1-squared. In general, in a multiple regression model, it is perfectly legitimate to have one or more of the k predictors that are mathematical functions of other predictors.

Tab 1.4: Experiment Analysis (1)



These are the data collected by Galileo during the Pisa experiment. The first column is just an index, the second column is the measured time and the third column the measured position. We will use this data to fit the parameters of the multiple linear regression defined previously.

```
1 df<-read.csv("galieodata.csv")
2 df$time2=df$time^2 #we make a new covariate
3 mymodel <- lm(position~time+time2 ,df)
4 mymodel
```

As said, since we are just considering multiple linear regression, we can use R to solve Galileo's problem, that means estimating the betas.

We can use ``lm'' (linear-model) from R to find the value of the betas coefficients. We read the dataset, we create a new column which includes time-squared (see the second line in the code box) and run linear regression.

Call:

1m(formula = position – time _time2, data = df)

Coefficients:

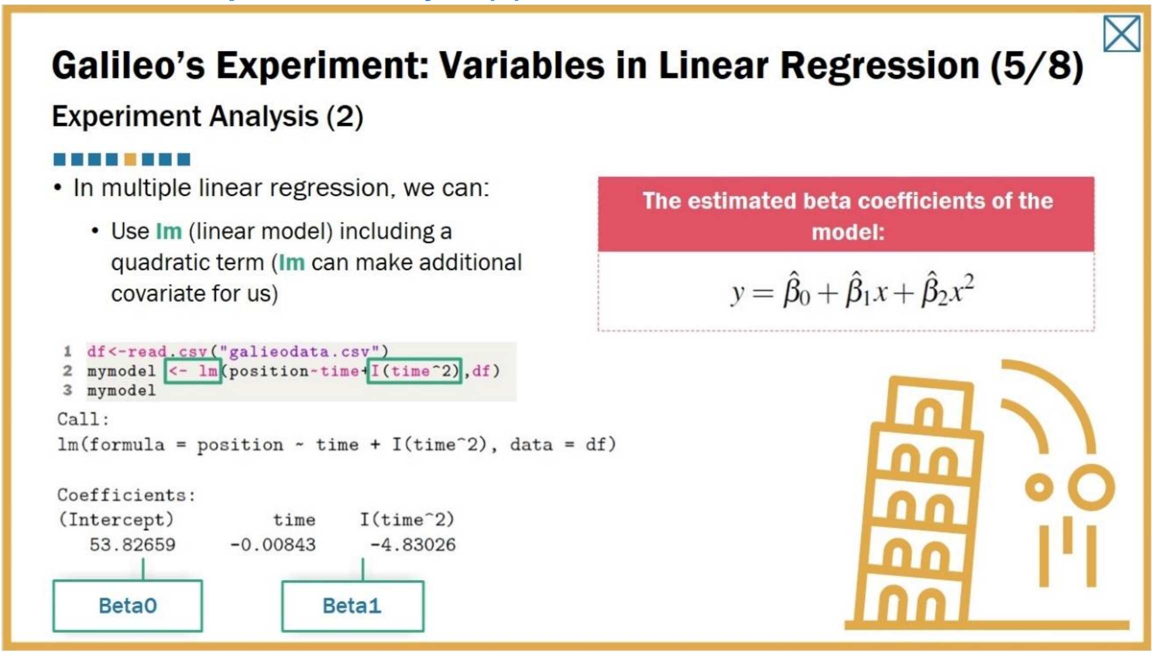
(intercept) time time2

53.82659 -0.00843 -4.83026

The estimated coefficients are equal to 53.8 (that is beta0), -0.008 (that is beta1) and -4.8 (that is beta2). These are the three estimated beta coefficients of the model you see in the equation.

We used hat to denote the estimated betas.

Tab 1.5: Experiment Analysis (2)



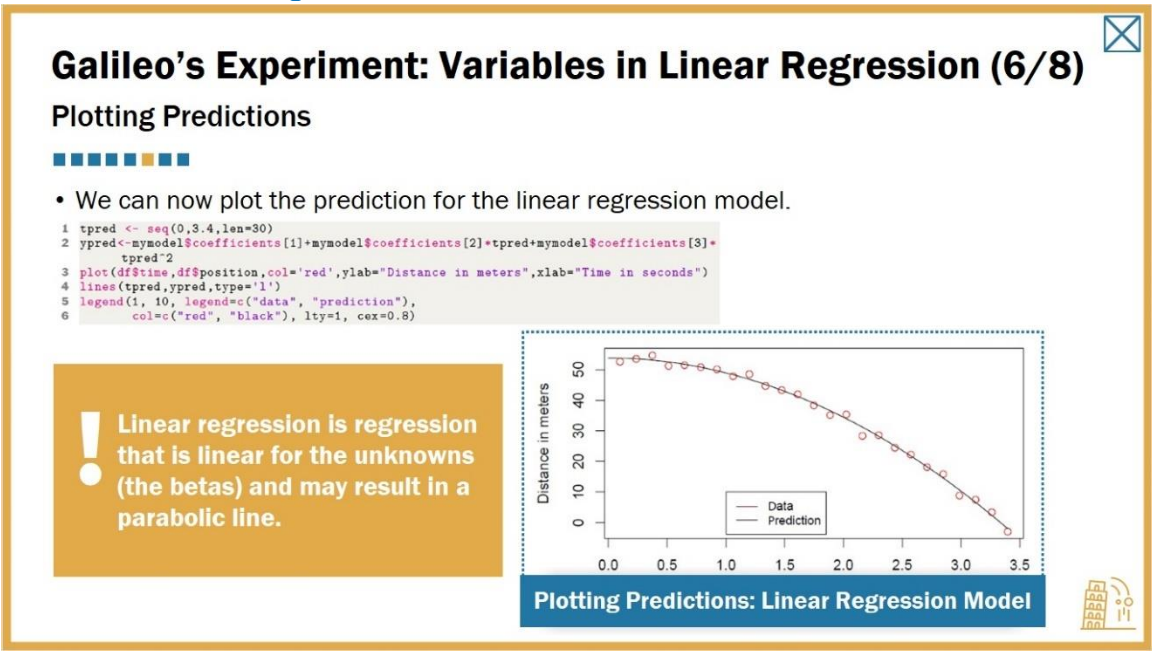
Equivalently, we can use ``lm'' (linear-model) including a quadratic term (``lm'' can make additional covariate for us)

```
df<-read.csv("galieodata.csv")
mymodel <- lm(position~time+I(time^2),df)
mymodel
```

Note the term I time-squared, which tells “lm” we want another covariate which is equal to time-squared.

The result, that is the output of “lm”, is obviously the same.

Tab 1.6: Plotting Predictions

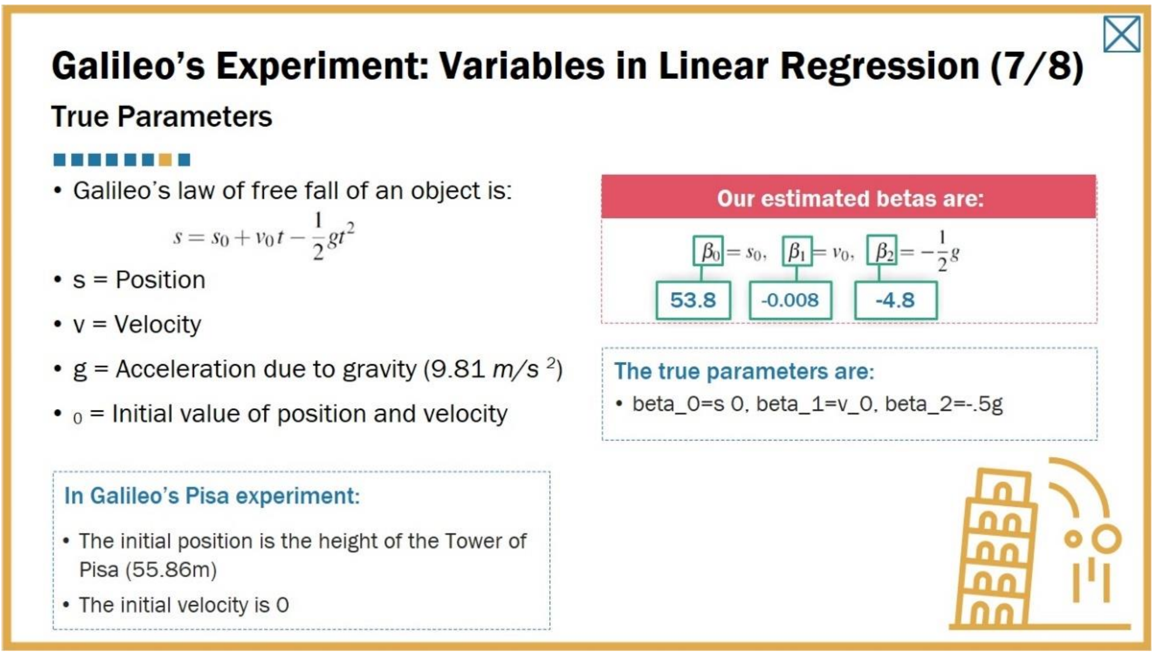


We can now plot the prediction for the linear regression model as shown in this code.

```
tpred <- seq(0,3.4,len=30)
ypred <-mymodel$coefficients [1]+ mymodel$coefficients
[2]*tpred+mymodel$coefficients [3]*
tpred^2
plot(df$time ,df$position ,col='red',ylab="Distance in meters",xlab="Time in seconds")
lines(tpred ,ypred ,type='l')
legend(1, 10, legend=c("data", "prediction"),
col=c("red", "black"), lty=1, cex=0.8)
```

You can notice that the continuous line is parabolic. Remember linear regression means regression which is linear with respect to the unknowns (the betas). So, it is perfectly legitimate that we obtain a line that is not “straight”

Tab 1.7: True Parameters



Thanks to Galileo today we know that the law for the free fall of an object is: space equal to initial position plus velocity times time minus .5 g time-squared

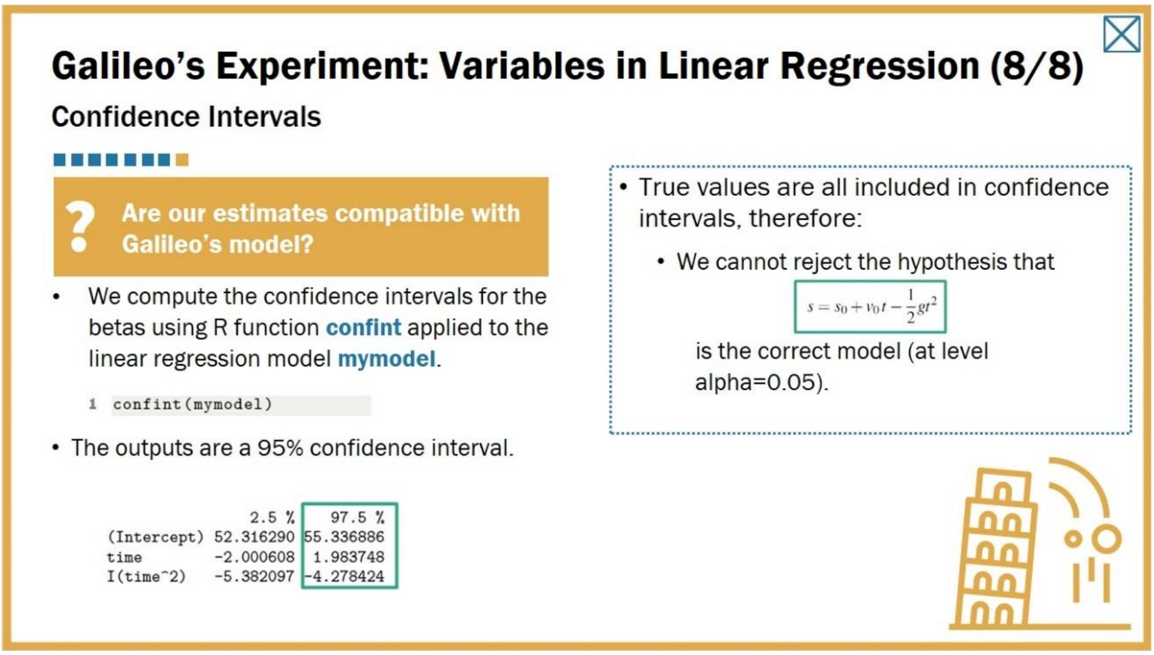
where s denotes position, v velocity, g is the acceleration due to gravity (9.81 m/s^2). The subscript 0 in the equation denotes the initial value of position and velocity.

In the Pisa experiment, the initial position is the height of the Tower of Pisa (55.86m)

and the initial velocity is 0 (it is a free fall)

We estimated 53.8 for beta0), -0.008 for beta1 and -4.8 for beta2, are these estimates compatible with the true parameters which are beta_0=s_0, beta_1=v_0, beta_2=-.5 g?

Tab 1.8: Confidence Intervals



Are our estimates compatible with Galileo's model?

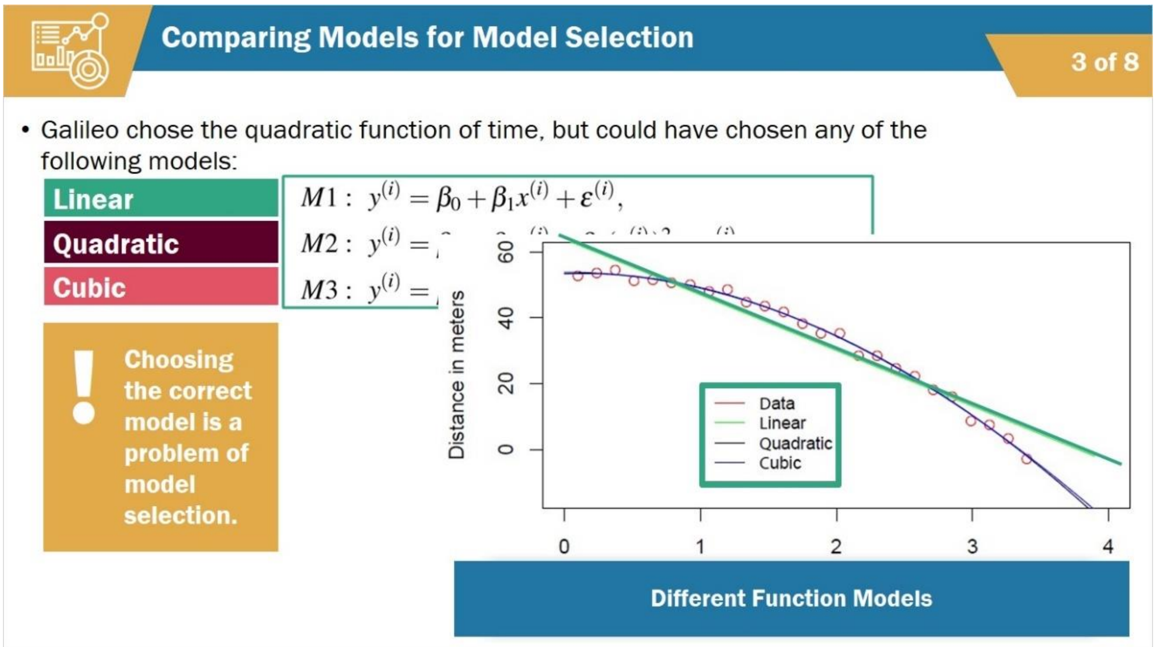
```
1 confint(mymodel)
```

We can answer this question by computing the confidence intervals for the betas through the R function "confint" applied to our linear regression model "mymodel".

The outputs are 95% confidence interval.

The table reports the 95% confidence interval for beta0 (intercept), beta1 (time) and beta2 (time squared). Note that the true values (55.86, 0, -4.905) are all included in the confidence intervals, which means we cannot reject the hypothesis that the model in the equation is the correct model (at level alpha=0.05). This is good.

Slide 3: Comparing Models for Model Selection

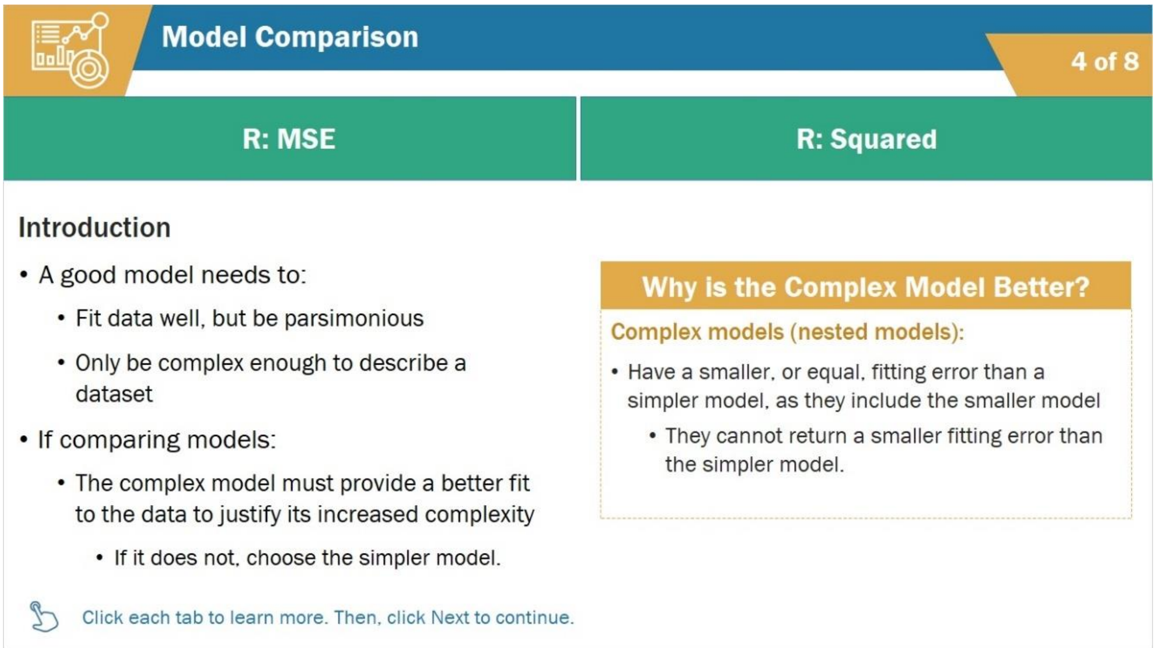


How did Galileo know that a quadratic polynomial is enough to fit the data?

If we look at these models, we see that Galileo could have chosen M1 which has no quadratic term or he could have chosen M2 which is a polynomial of degree 2. Alternatively, he could have considered M3 which includes a cubic term.

Choosing the correct model is a problem of model selection. From the graph, you can see that M1 (linear) does not fit the data well, while M2(quadratic) and M3 (cubic) both fit the data.

Slide 4: Model Comparison



A good model needs to fit data well, but it also needs to be parsimonious.

A good model should be only be as complex as necessary to describe the data.

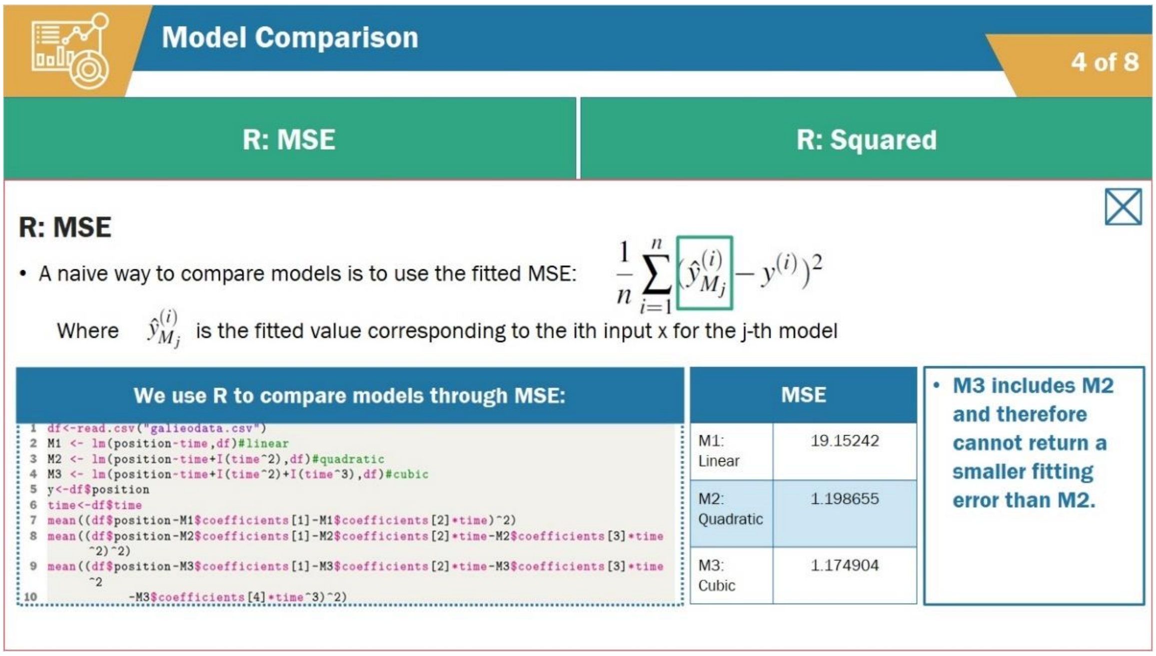
If we are comparing a simple model with a complex model, the complex model needs to provide a much better fit to the data in order to justify its increased complexity. If it cannot, then the more simpler model should be preferred.

For nested models, meaning one model contains a subset of the predictors of the larger model, the complex model will always have a smaller (or equal) fitting error than a simpler model.

Intuitively, this holds because the complex model includes the smaller model and so it cannot return a smaller fitting error than the simpler model.

Click each tab to learn more about ways to compare models. When you are ready, click next to continue.

Tab 1: **R: MSE**



A naive way to compare models is by using the fitted Mean Squared Error:

where y-hat-m_j denote the prediction for model M_j at the time t_j. That is, it is

the fitted value corresponding to the input x_i for model M_j.

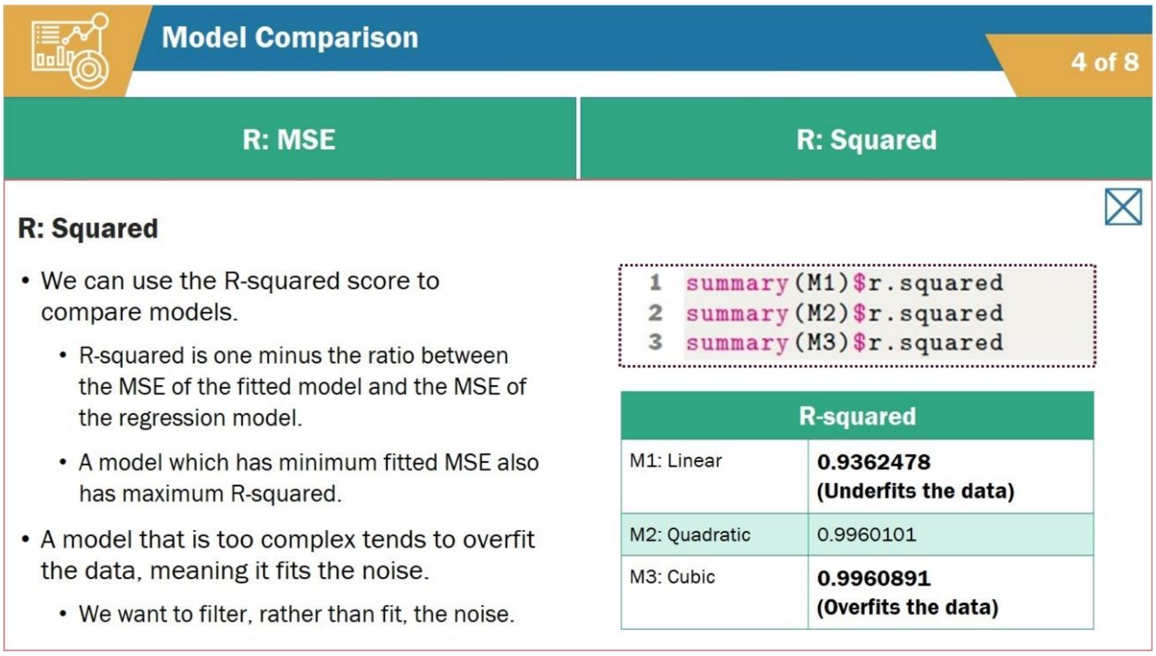
We can use R to compare the models through MSE:

The MSE of the three models are M1 (linear):19.15, M2 (quadratic):1.19, M3 (cubic) 1.17.

You can see that MSE for M3 is smaller, so the motion is not parabolic according to the minimum fitted MSE criterion.

We know that this is wrong: it is parabolic. The issue here is that a M3 is always going to have a smaller fitted MSE than M2. The complex model M3 includes the smaller model M2 and so it cannot return a smaller fitting error than M2l.

Tab 2: **R-Squared**



The conclusion that M3 is the best model is also true if we consider the R-Squared score to compare models. R-Squared is 1 minus the ratio between the MSE of the fitted model and the MSE of the regression model which does not use any predictors. Therefore, a model which has minimum fitted MSE has also maximum R-Squared, as we can see in the slide

The model with the highest R-squared is again M3. However, this model is too complex,

A too complex model tends to overfit the data, which means it fits the noise (which previously we denoted by epsilon). This means it is not a good model: we want to filter the noise (measurement error) not to fit it.

```
1    summary (M1) $r.squared
2    summary (M2) $r.squared
3    summary (M3) $r.squared
```

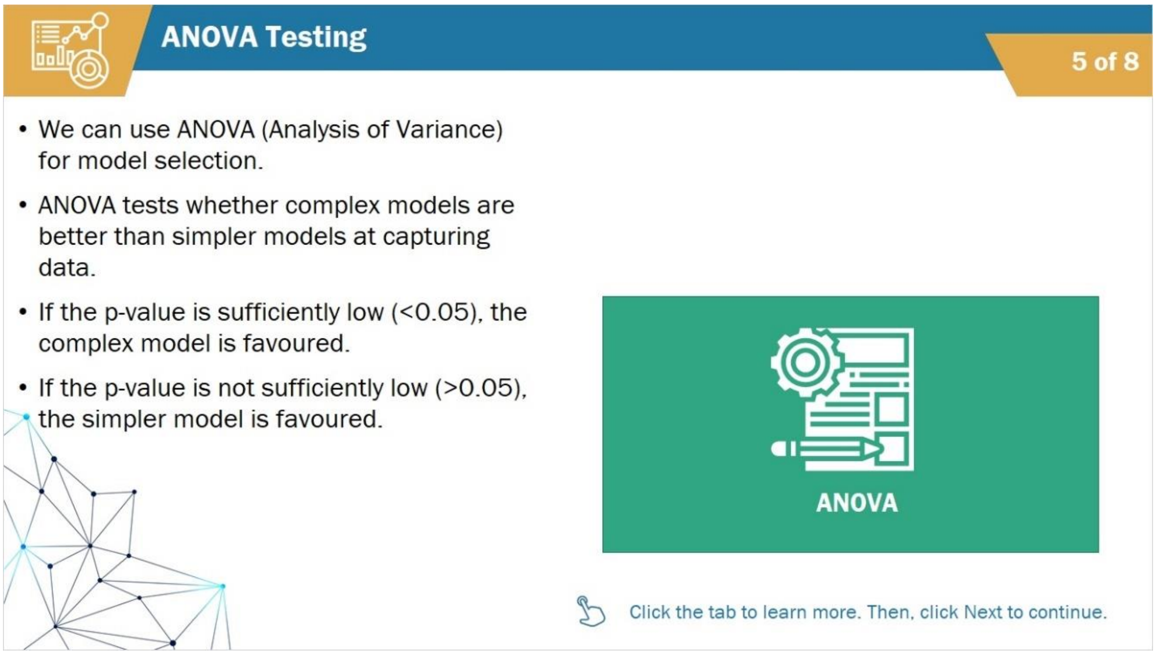
```
[1]    0.9362478

[1]    0.9960101

[1]    0.9960891
```

M3 overfits the data, while M1 clearly underfits it.

Slide 5: **Anova Testing**



For model selection, we can use ANOVA (Analysis of Variance)

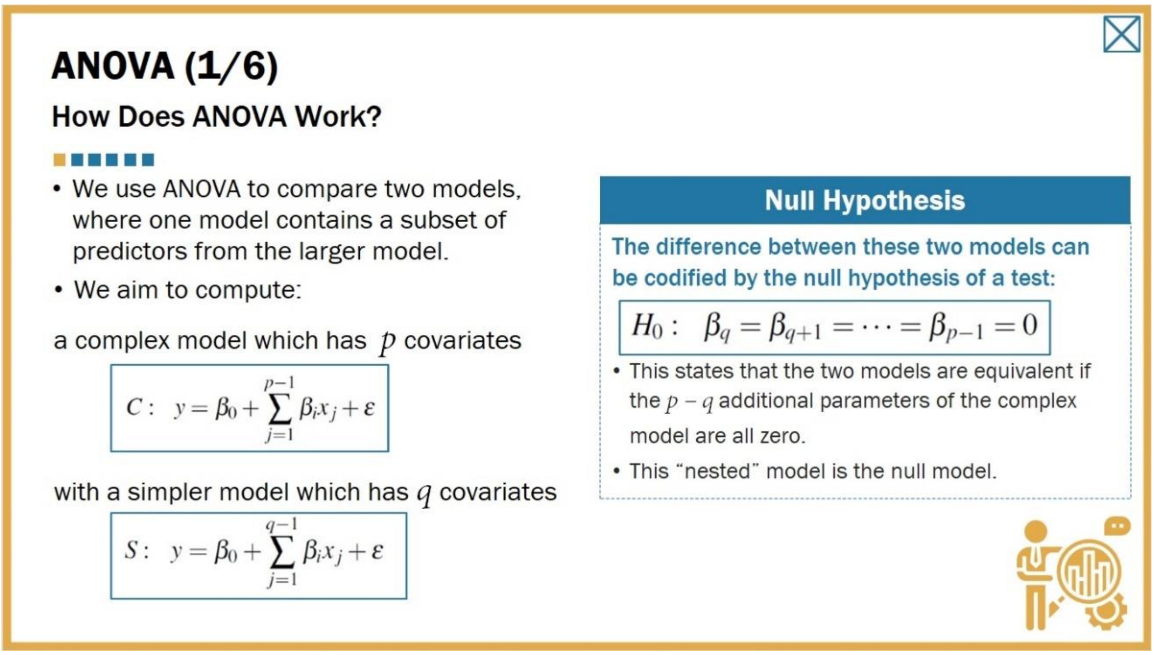
ANOVA testing aims to test whether the more complex model is significantly better at capturing the data than the simpler model.

If the p-value resulting from the ANOVA test is sufficiently low (usually less than 0.05), we conclude that the more complex model is significantly better than the simpler model, and thus favour the more complex model.

If the p-value is not sufficiently low (usually greater than 0.05), we should favour the simpler model

Click the tab to learn more about ANOVA. When you are ready, click next to continue.

Tab 1.1: ANOVA



We use ANOVA to compare two models, this only works when one model is ``nested'' inside the other, meaning one model contains a subset of the predictors from only the larger model.

So, we aim to compute a complex model denoted as C which has p-covariates, with a simpler model denoted as S, which has q-covariates.

The difference between these two models can be codified by the null hypothesis of a test, denoted as H_0 which states that the two models are equivalent. This happens if the p-q additional parameters of the complex model are all zero.

In other words, the beta-parameters from the full model that are not in the null model are zero. The resulting model, which is therefore nested, is the null model.

Tab 1.2: F-statistic

ANOVA (2/6)

F-statistic

- We denote the fitted values for model C as: $\boxed{\hat{y}_C^{(i)}}$
- The fitted values for model S as: $\boxed{\hat{y}_S^{(i)}}$
- We then construct the following F-test statistic:

$$F = \frac{\frac{\sum_{i=1}^{n}(y_C^{(i)} - \hat{y}_S^{(i)})^2}{p-q}}{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_C^{(i)})^2}{n-p}}$$

Numerator	• Compares the sum of the squared differences of the fitted values
Degrees of freedom (p-q)	• The difference of the number of beta-parameters estimated between the two models
Denominator	• The fitting error of the complex model divided by the degrees of freedom (n-p)

We denote the fitted values for model C as y_hat_C_i where I is the i-th observation and the fitted values for model S as y_hat_S_i

We can then construct the F-test statistic described in the equation, which is the ratio of two fractions.

The numerator of the top fraction compares the sum of the squared differences of the fitted values for the two models. The degrees of freedom (p-q) is the difference of the number of beta-parameters estimated between the two models.

The numerator of the bottom fraction is instead the fitting error of the complex model divided by the degrees of freedom n-p.

Tab 1.3: F-test

ANOVA (3/6)

F-test

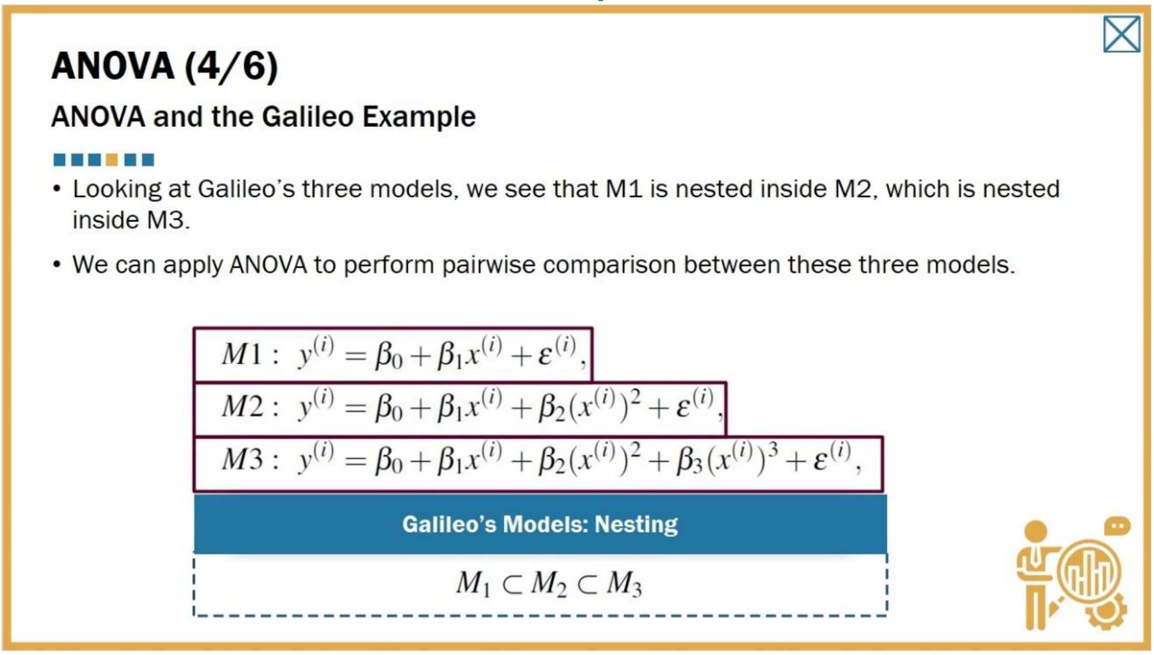
- The numerator represents:
  - The **between-group variability**
  - The denominator of the **within-group variability**
- F has an F-distribution with p-q and n-p degrees of freedom.
  - We can compute the p-value for the null-hypothesis using this distribution.

$$F = \frac{\boxed{\frac{\sum_{i=1}^{n}(y_C^{(i)} - \hat{y}_S^{(i)})^2}{p-q}}}{\boxed{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_C^{(i)})^2}{n-p}}}$$

F-test Statistic

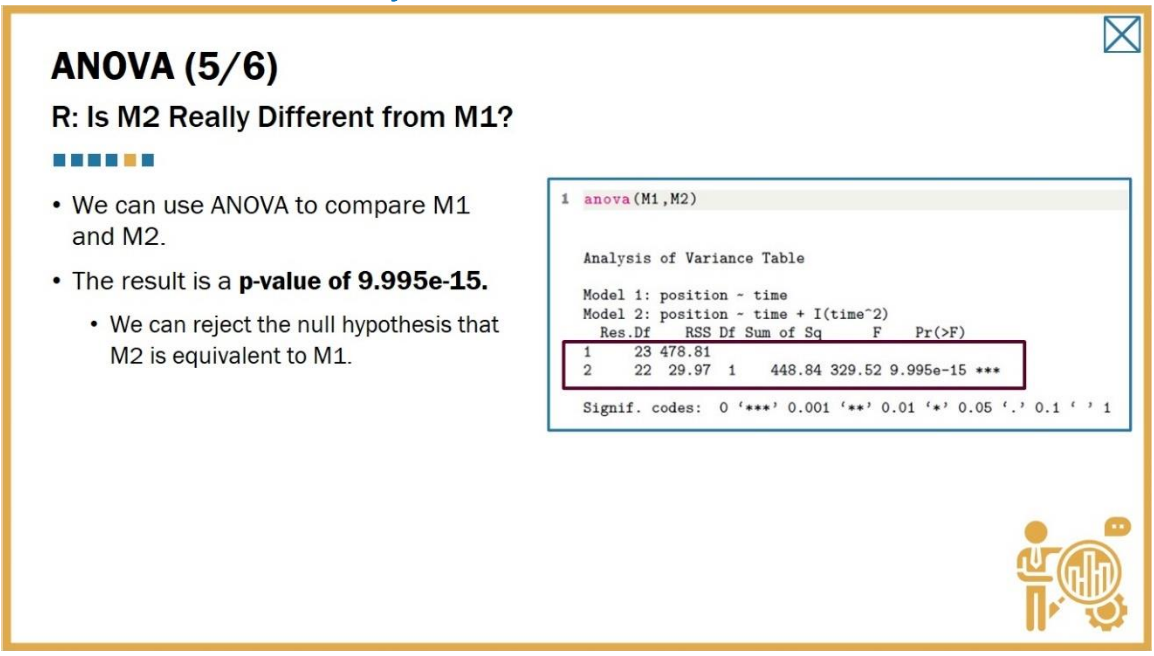
Therefore, the top fraction represents the *between-group variability* and the bottom fraction the *within-group variability*. F has a F-distribution with p-q and n-p degrees of freedom and, therefore, we can compute the p-value for the null-hypothesis using the F-distribution.

Tab 1.4: ANOVA and the Galileo Example



Let's go back to Galileo experiment, where we had three models M1 (linear), M2 (quadratic) and M3 (cubic). We use ANOVA to perform pairwise comparisons between these three models, where one model is ``nested" inside the other, meaning one model contains a subset of the predictors from only the larger model. Here we see our three models, Looking at these equations, we can see that model M1 is nested into M2 which is nested into M3, so we can apply Anova to compare these 3 models.

Tab 1.5: R: Is M2 Really Different from M1?

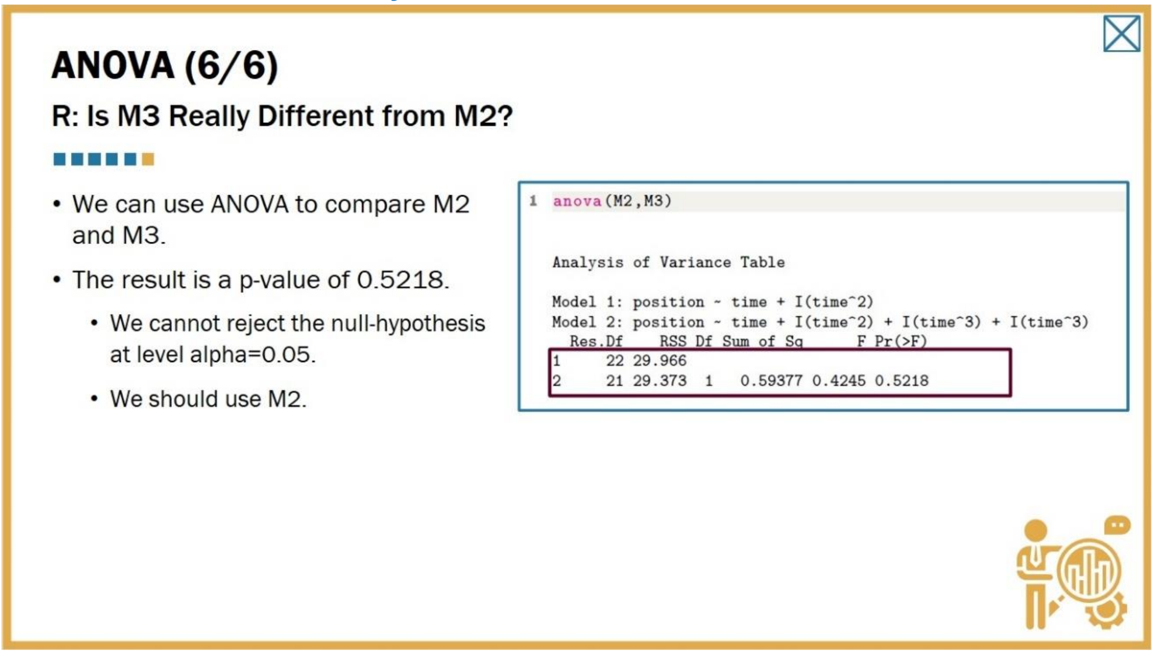


We can use anova to compare M1 and M2, and the result is a p-value of 9.995e-15:

```
1 anova(M1,M2)
```

So yes, we can reject the null hypothesis that M2 is equivalent to M1.

Tab 1.6: R: Is M3 Really Different from M2?

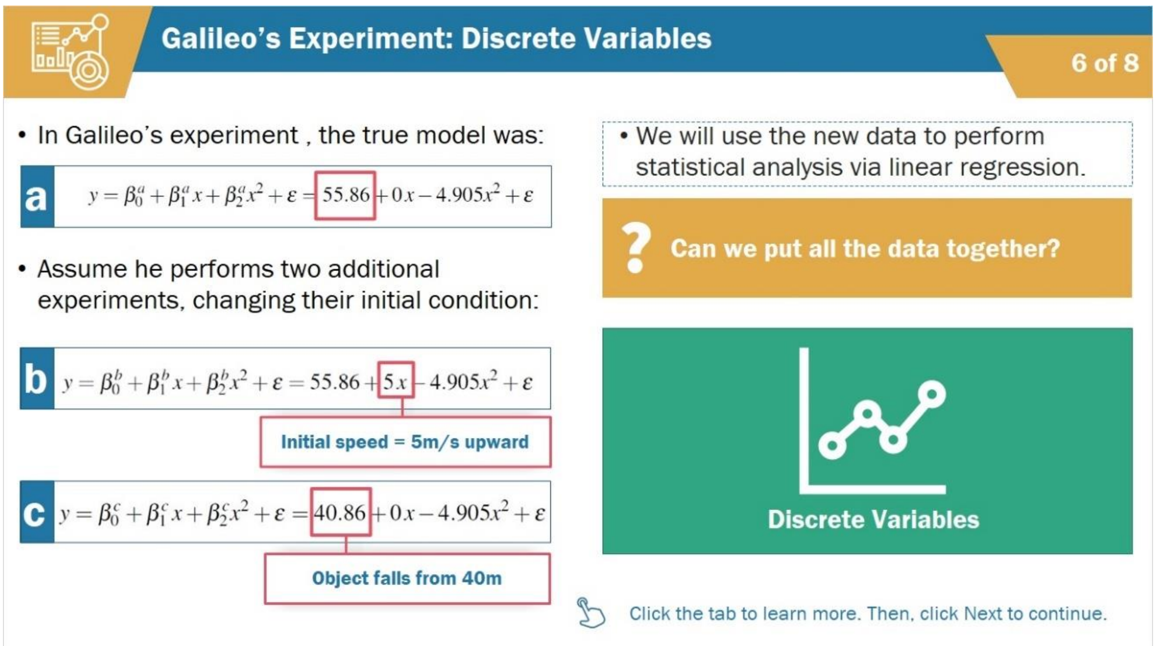


We can use anova to compare them, and the result is a p-value of 0.5218:

```
1 anova(M2,M3)
```

In this case, we cannot reject the null-hypothesis at level alpha=0.05, so we should use M2.

Slide 6: Galileo's Experiment: Discrete Variables



Let's now change a bit topic. In the previous Galileo experiment (denoted as a), the true model was the one you see in the equation, where 55.86 is the height of the Pisa's tower

Let's assume now that Galileo decides to perform two additional experiments by changing the initial condition. In the second experiment (denoted by b) he gives an initial speed (5m/s upward) to the falling object.

In the third experiment (denoted as c), he lets the object fall from 40m (instead from the top of the tower)

Our goal is to use the new data (discrete variables) to perform again the statistical analysis via linear regression.

It is clear we can do that by simply using the data from experiment b and then the data from experiment c, but can we put all data together?

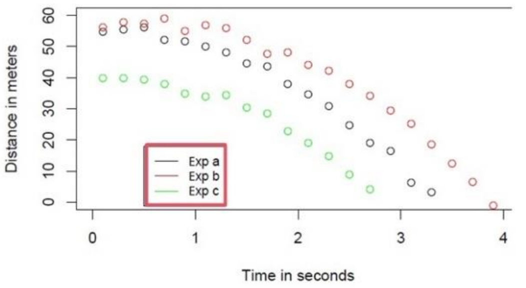
Using these three experiments, we will explore the topic of discrete variables in more detail.

Click the tab to learn more about discrete variables, when you are ready, click next to continue.

Tab 1.1: Discrete Variables

Discrete Variables (1/7)

Why Do We Want to Put All the Data Together?



Galileo's Experiments

Why put all the data together?

a $y = \beta_0^a + \beta_1^a x + \beta_2^a x^2 + \varepsilon = 55.86 + 0x - 4.905x^2 + \varepsilon$

b $y = \beta_0^b + \beta_1^b x + \beta_2^b x^2 + \varepsilon = 55.86 + 5x - 4.905x^2 + \varepsilon$

c $y = \beta_0^c + \beta_1^c x + \beta_2^c x^2 + \varepsilon = 40.86 + 0x - 4.905x^2 + \varepsilon$

- The magnitude (variance) of the measurement error is the same, as we are measuring the position in the same way.
- The coefficient which multiples x-square is the same.
- Model a = model c + 10 (only initial position changes).
- Model a = model b - x*5 (we started with a non-zero initial speed).

We can see the data of the three experiments in this scatter plot: the original experiment is in black, the one with a non-zero initial velocity is in red and the one from 40m is in green.

Looking at the three model equations, we can see that:

- The magnitude (variance) of the measurement error is the same, since we measure the position in the same way (so same errors).

-The coefficient which multiples x-square is also the same (related to gravity).

- Model a is equal to Model c plus 10, because we only change the initial position.

-Model a is equal to model b minus x*5, because we started with a non zero initial speed.

Tab 1.2: A Single Model

Discrete Variables (2/7)

A Single Model

- Using indicator variables, we can write all three models as a single equation:

$$y = \boxed{55.86 I_{\{a,b\}}}(e) + \boxed{40.86 I_{\{c\}}}(e) + \boxed{5xI_{\{b\}}}(e) - \boxed{4.905x^2} + \varepsilon$$

where e denotes the experiment (a, b, c) and

$$I_A(e)$$

is an indicator function which is equal to one when e is included in the set A and zero otherwise.

- We can equivalently rewrite the model as:

$$y = \boxed{55.86 I_{\{a\}}}(e) + 55.86 I_{\{b\}}(e) + \boxed{40.86 I_{\{c\}}}(e) + \boxed{5xI_{\{b\}}}(e) - \boxed{4.905x^2} + \varepsilon$$

so that the set A in the indicators is always a singleton.

Using indicator variables, we can write all the three models as a single equation, see equation where “e” denotes the experiment. “e” can assume values a,b,c. I_{A}(e) is an indicator function which is equal to one when e is included in the set A and zero when e is not included in A.

So, the first equation tells us that we have the term 55.86 for models a and b. We have 40.86 for model c. The term 5 times x is only present for model b. Conversely, the quadratic term (-4.905x^2) is present for all experiments.

We can equivalently rewrite the model as we see in this last equation, so that the set A in the indicators is always a singleton.

Tab 1.3: A Single Model and Interaction Term

Discrete Variables (3/7)

A Single Model and Interaction Term

- In a real experiment, we do not know the coefficients, therefore:

$$\boxed{y = \beta_0 I_{\{a\}}(e) + \beta_1 I_{\{b\}}(e) + \beta_2 I_{\{c\}}(e) + \beta_3 x + \beta_4 x I_{\{b\}}(e) + \beta_5 x^2 + \varepsilon}$$

- This is a multiple linear regression model where the covariates are:

$$[I_{\{a\}}(e),\ \ I_{\{b\}}(e),\ \ I_{\{c\}}(e),\ \boxed{x,}\ \boxed{xI_{\{b\}}(e)},\ \boxed{x^2]}$$

- The first three covariates are binary variables.
- The product of x and the indicator (i. e. the product of two covariates) is called the **interaction term**.

In a real experiment we do not know the coefficients and so we need to replace the numbers with variables: the betas representing the unknowns.

We have again a multiple linear regression model where the covariates are the three indicators, x, x-squared and the product of x times an indicator of the experiment b

The first three covariates are binary variables: they can be either 1 or 0. The product of x and the indicator, which is the product of two covariates, is called interaction term (fifth term in the list).

This is a general name, any product of covariate is called the interaction term.

Tab 1.4: Building Covariates

Discrete Variables (4/7)

Building Covariates

$$[I_{\{a\}}(e),\ \ I_{\{b\}}(e),\ \ I_{\{c\}}(e),\ \ x,\ \ xI_{\{b\}}(e),\ \ x^2]$$

? How do we use linear regression to estimate the coefficients of the previous model?

- To build the covariates, we need to transform the data by adding:
  - The indicators
  - The interaction term
  - The quadratic term

Galileo's Data

e	time	position
a	0.1	54.9140855
b	0.1	56.3468067
c	0.1	39.9084155
a	0.3	55.6038492
b	0.3	57.9318287
c	0.3	39.8599381
a	0.5	56.2228453
b	0.5	57.5672652
c	0.5	39.3884874
.	...	...

Indicators and Other Covariates

I_a	I_b	I_c	time	time*I_b	time^2	position
1	0	0	0.1	0	0.01	54.9140855
0	1	0	0.1	0.1	0.01	56.3468067
0	0	1	0.1	0	0.01	39.9084155
1	0	0	0.3	0	0.09	55.6038492
0	1	0	0.3	0.3	0.09	57.9318287
0	0	1	0.3	0	0.09	39.8599381
1	0	0	0.5	0	0.25	56.2228453
0	1	0	0.5	0.5	0.25	57.5672652
0	0	1	0.5	0	0.25	39.3884874
...						

How do we use linear regression to estimate the coefficients of the linear regression model which has the six covariates listed in the previous slide?

e	time	position
a	0.1	54.9140855
b	0.1	56.3468067
c	0.1	39.9084155
a	0.3	55.6038492
b	0.3	57.9318287
c	0.3	39.8599381
a	0.5	56.2228453
b	0.5	57.5672652
c	0.5	39.3884874

We have the following data: the first column is an identifier variable denoting the experiment (a,b,c), the second column (time) and the third column (position) include the data of the three experiments all stacked together. For each row, the first column “e” identifies to which experiment the corresponding pair (time, position) belongs to.

To build the covariates we need to transform the data by adding the indicators for the three experiments, the interaction term and the quadratic term, seen here in the second table:

Tab 1.5: R: Encoding

Discrete Variables (5/7)

R: Encoding

- The procedure to transform a categorical variable (a, b, c) into the three indicators is called **one-hot encoding**.
- In R, we use the function “one-hot”.

```
library(mltools)
library(data.table)
encoding <-one_hot(as.data.table(df$experiment))
encoding$time<-df$time
encoding$timexIb<-df$time*encoding[,2]
encoding$time2<-df$time^2
encoding$position<-df$position
encoding
```

- We add the interaction and quadratic term to give the dataset:

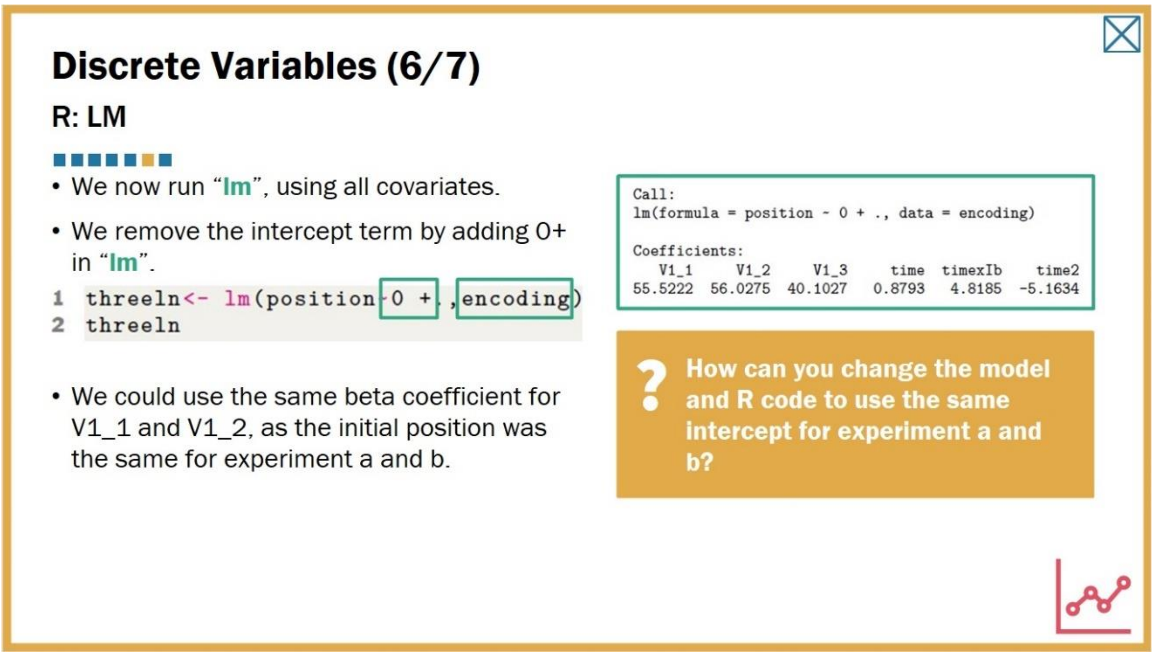
	V1_1	V1_2	V1_3	time	timexIb	time2	position
1:	1	0	0	0.1	0.0	0.01	54.9140855
2:	0	1	0	0.1	0.1	0.01	56.3468067
3:	0	0	1	0.1	0.0	0.01	39.9084155
4:	1	0	0	0.3	0.0	0.09	55.6038492
5:	0	1	0	0.3	0.3	0.09	57.9318287
6:	0	0	1	0.3	0.0	0.09	39.8599381
7:	1	0	0	0.5	0.0	0.25	56.2228453
8:	0	1	0	0.5	0.5	0.25	57.5672652
9:	0	0	1	0.5	0.0	0.25	39.3884874

....

The procedure to transform a categorical variable (a,b,c) into the three indicators is called one-hot encoding and we can use R to do that using the function “one_hot”. We also add the interaction and quadratic term leading to the dataset you see here.
The three indicators have been called V1_1, V1_2 and V1_3.

```
library(mltools)
library(data.table)
encoding <-one_hot(as.data.table(df$experiment))
encoding$time<-df$time
encoding$timexlb <-df$time*encoding[,2]
encoding$time2 <-df$time^2
encoding$position <-df$position
encoding
```

Tab 1.6: R: Lm

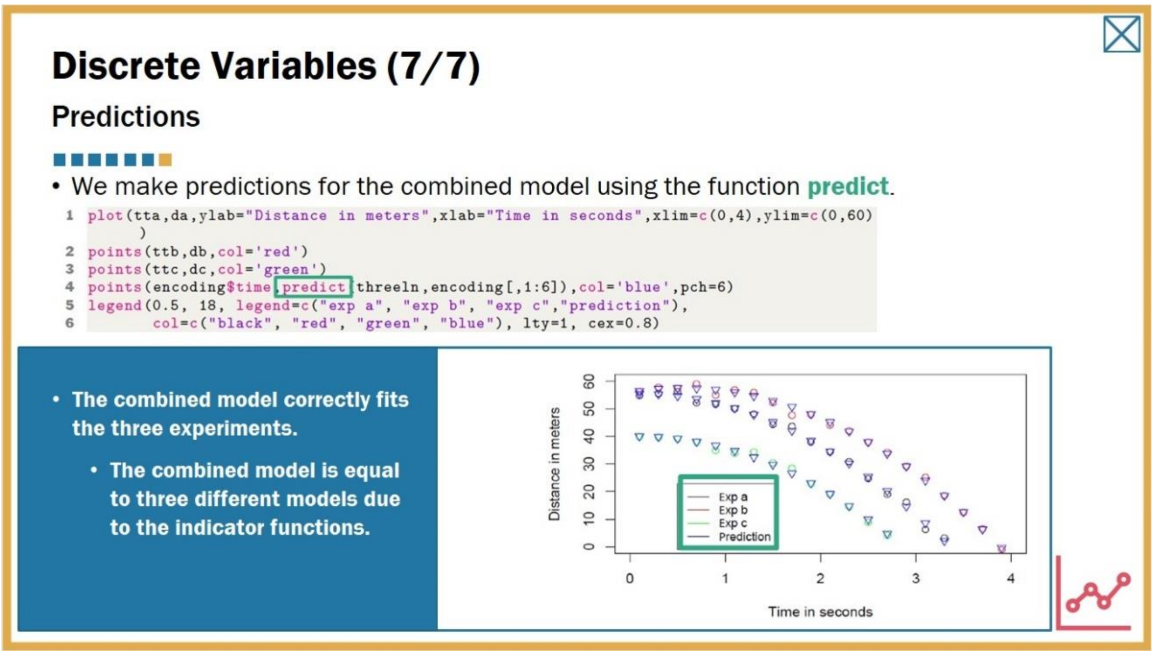


We can now run `lm` using all covariates. We need to remove the intercept term because we have already included it through the indicators covariates. To remove the intercept, we simply add 0+ in `lm`. The dot means we include all covariates inside the dataframe encoding.

```
threeln <- lm(position~0 +.,encoding)
threeln
```

You can see from the result shown here that the estimated betas are close to the true value (you can compare their values with the true ones). We could have the same beta coefficient for V1_1 and V1_2 because we know that the initial position was the same for experiment a and b; this can easily be changed (**How can you change the model and the R code to use the same intercept for experiment a and b.** ? Try yourself!

Tab 1.7: Predictions

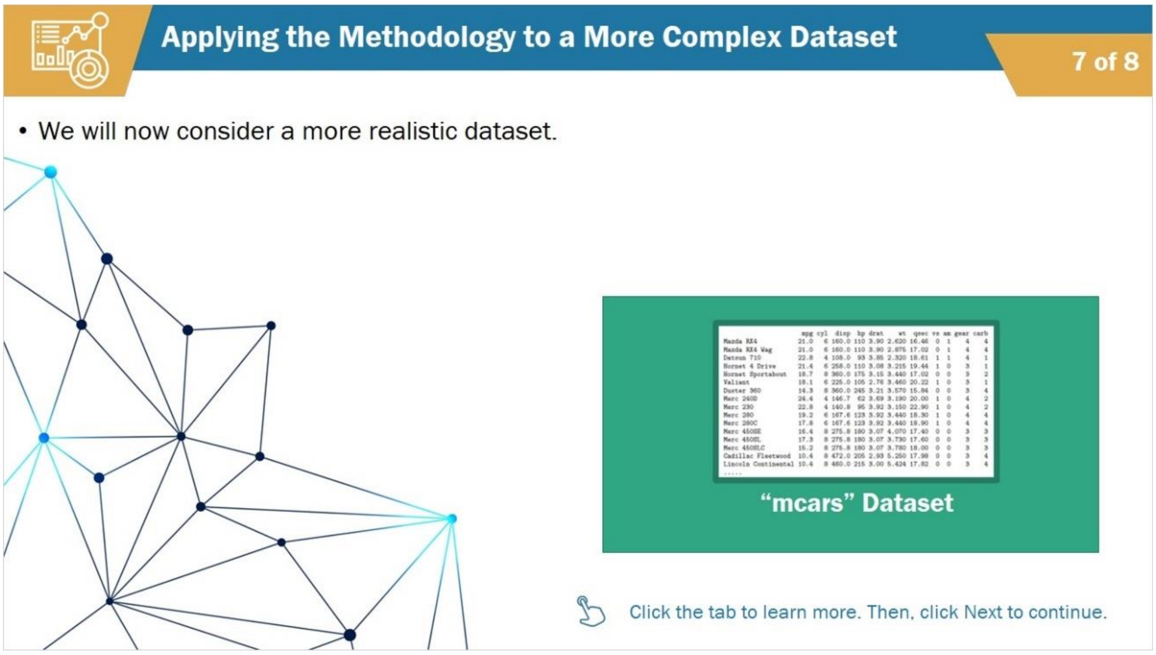


We can make predictions for the combined model using the function `predict`, see the code.

```
plot(tta ,da,ylab="Distance in meters",xlab="Time in seconds",xlim=c(0,4),ylim=c(0,60))
points(ttb ,db,col='red')
points(ttc ,dc,col='green')
points(encoding$time ,predict(threeln ,encoding [,1:6]),col='blue',pch=6)
legend(0.5, 18, legend=c("exp a", "exp b", "exp c","prediction"),
col=c("black", "red", "green", "blue"), lty=1, cex=0.8)
```

Note on the graph how the combined model correctly fit the three experiments. This is possible because, due to the indicators functions, the combined model is equal to three different models (sharing some of the coefficients).

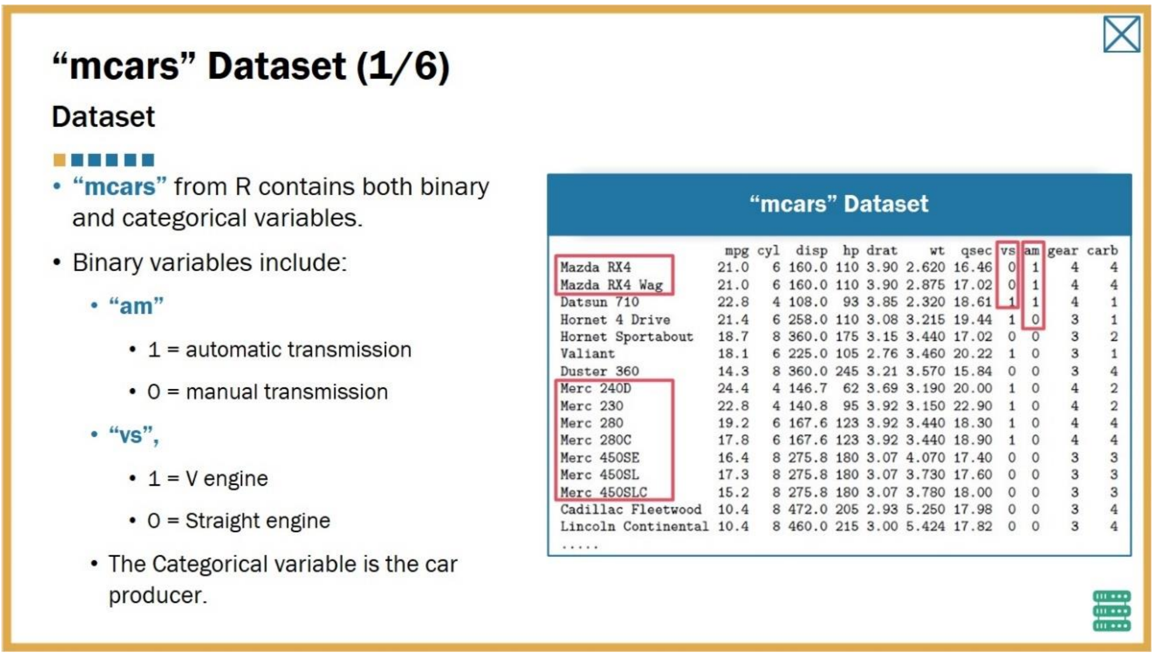
Slide 7: Applying the Methodology to a More Complex Dataset



The approach we have just considered (powers, indicators, interaction) is very general. We are now going to examine a more realistic dataset.

We will consider the "mcars" dataset from R. Click the tab to learn more. When you are ready, click next to continue.

Tab 1.1: "mcars" Dataset



	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
.....											

The "mcars" includes binary variables: ``am'' (where 1 means the car has automatic transmission, 0 means manual transmission) and ``vs'' (1 means the car has a V engine; 0 means it has a straight engine. It has also a categorial variable denoting the car-producer (Mazda, Merc,...) in the index.

Tab 1.2: Variables

"mcars" Dataset (2/6)

Variables

- We are interested in three variables:
  - mpg (fuel efficiency, in miles per gallon)
  - hp (horsepower, in foot-pound per second)
  - am (transmission: automatic or manual)

We are interested in mpg as the response output variable.

"mcars" Dataset

	mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
Mazda RX4	21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
Mazda RX4 Wag	21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
Datsun 710	22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
Hornet 4 Drive	21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
Hornet Sportabout	18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
Valiant	18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
Duster 360	14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
Merc 240D	24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
Merc 230	22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
Merc 280	19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4
Merc 280C	17.8	6	167.6	123	3.92	3.440	18.90	1	0	4	4
Merc 450SE	16.4	8	275.8	180	3.07	4.070	17.40	0	0	3	3
Merc 450SL	17.3	8	275.8	180	3.07	3.730	17.60	0	0	3	3
Merc 450SLC	15.2	8	275.8	180	3.07	3.780	18.00	0	0	3	3
Cadillac Fleetwood	10.4	8	472.0	205	2.93	5.250	17.98	0	0	3	4
Lincoln Continental	10.4	8	460.0	215	3.00	5.424	17.82	0	0	3	4
.....											

Assume we are interested in three of the variables:

- mpg: fuel efficiency

- hp: horsepower

- am: transmission. Automatic or manual.

In particular, we are interested in mpg as the response (output) variable.

Tab 1.3: Model

"mcars" Dataset (3/6)

Model

Assume we are considering the model:

$$\boxed{mpg = \beta_0 + \beta_1\, hp + \beta_2\, am}$$

Since "am" is a binary variable, we are considering two models.

Model 1:	Model 2:
Cars with automatic transmission (am=1) $mpg = \beta_0 + \beta_2 + \beta_1\, hp$	Cars with manual transmission (am=0) $mpg = \beta_0 + \beta_1\, hp$

- Putting these models together using am allows us to exploit the "hp" in both models so we can use a unique coefficient (beta1) for "hp".

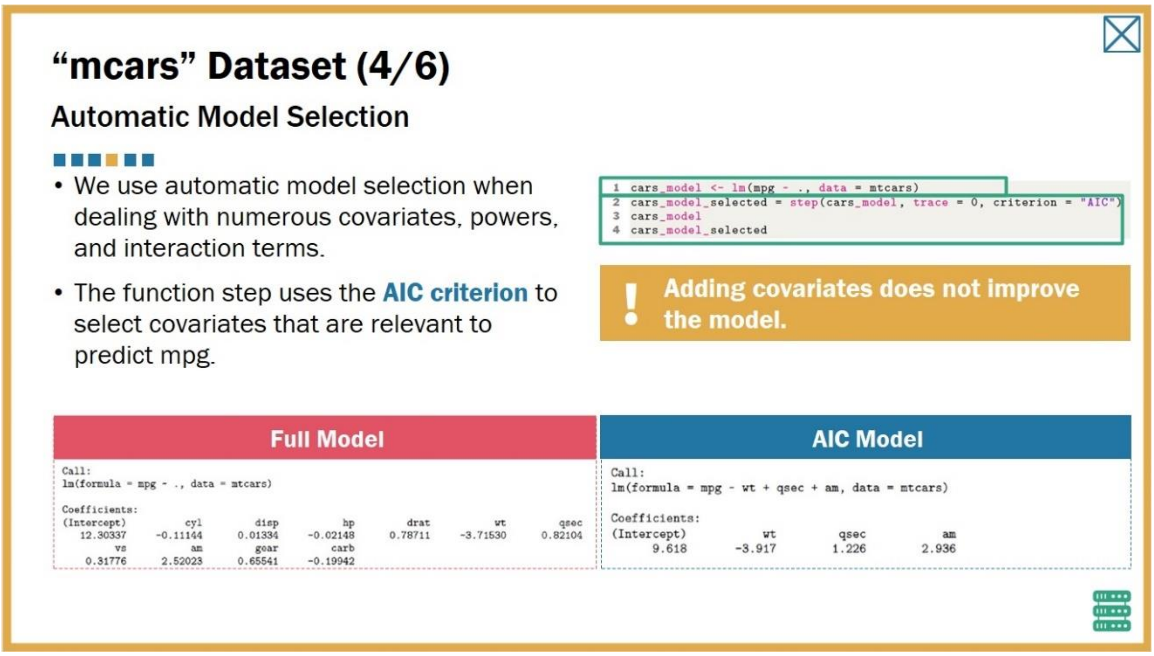
Assume we consider the model in the equation. Since "am" is a binary variable, we are basically considering two models.

The first model corresponds to cars with automatic transmission (where am=1). Note that the intercept is beta0 plus beta2

The first model corresponds to cars with Manual transmission (when am=0)

Putting the models together using am as indicator variable allows us to exploit that “hp” is in both models and so we use a unique coefficient (beta1) for “hp”.

Tab 1.4: Automatic Model Selection



Consider the full number of covariates in mcars. How can we choose the model when we have so many covariates, considering we can also add powers and interaction terms?

We can use automatic model selection. The function step from R uses the Akaike Information Criterion, (AIC), which we will discuss later, to select the covariates that are relevant to predict mpg.

We can run AIC using the following commands in R. Note the difference between the full-model denoted as cars_model, which uses all-covariates, and the model selected using AIC, denoted as cars_model_selected.

1 cars_model <- lm(mpg ~ ., data = mtcars)

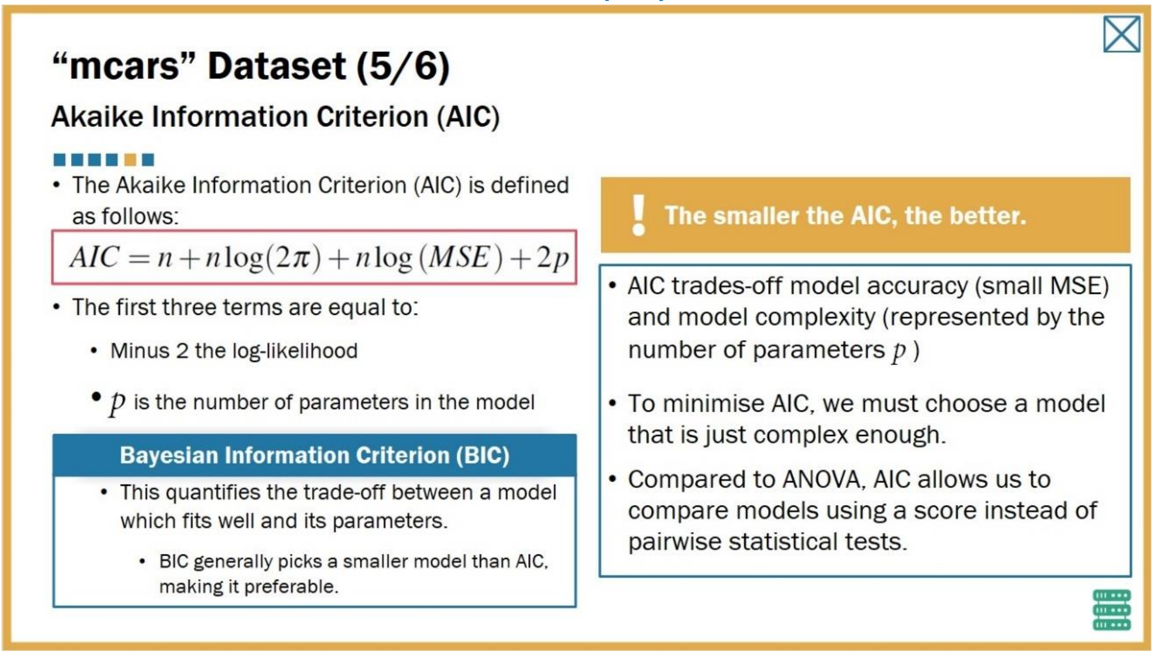
2 cars_model_selected = step(cars_model, trace = 0, criterion = “AIC”)

3 cars_model

4 cars_model_selected

The latter uses only three covariates: wt , qsec and am. It means that adding the other covariates does not improve the model. I am now going to explain what AIC is and how the function “step” works.

Tab 1.5: Akaike Information Criterion (AIC)



What is the Akaike Information Criterion? AIC for short is defined as in the equation

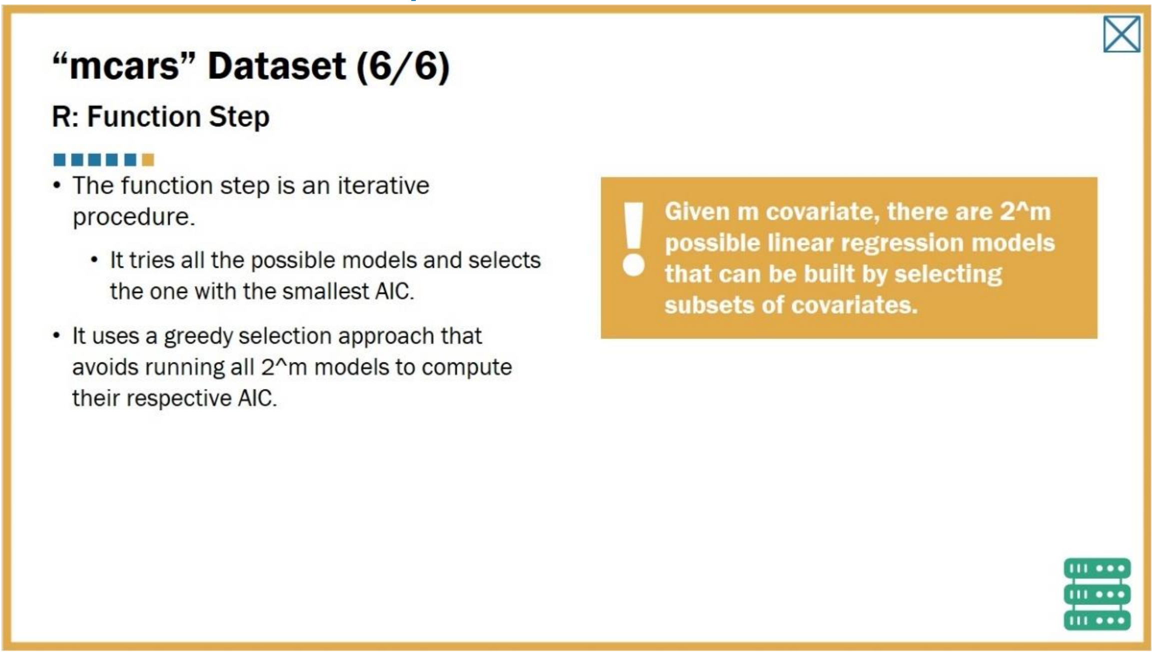
The smaller the AIC, the better. The first three terms are equal to: minus 2 the log-likelihood. p is instead the number of parameters of the model. Therefore, AIC trades-off model accuracy (small MSE) and model complexity (represented by the number of parameters p).

We have seen that by increasing p we always decrease MSE. So, to minimize AIC, we must choose a model which is complex enough but not too complex.

Compared to anova, AIC allows us to compare models using a score instead of a pairwise statistical tests.

There are other criteria, similar to AIC, we can use instead of AIC, for instance Bayesian Information Criterion (BIC). BIC also quantifies the trade-off between a model which fits well and the number of model parameters, however for a reasonable sample size, generally picks a smaller model than AIC, which is usually preferable.

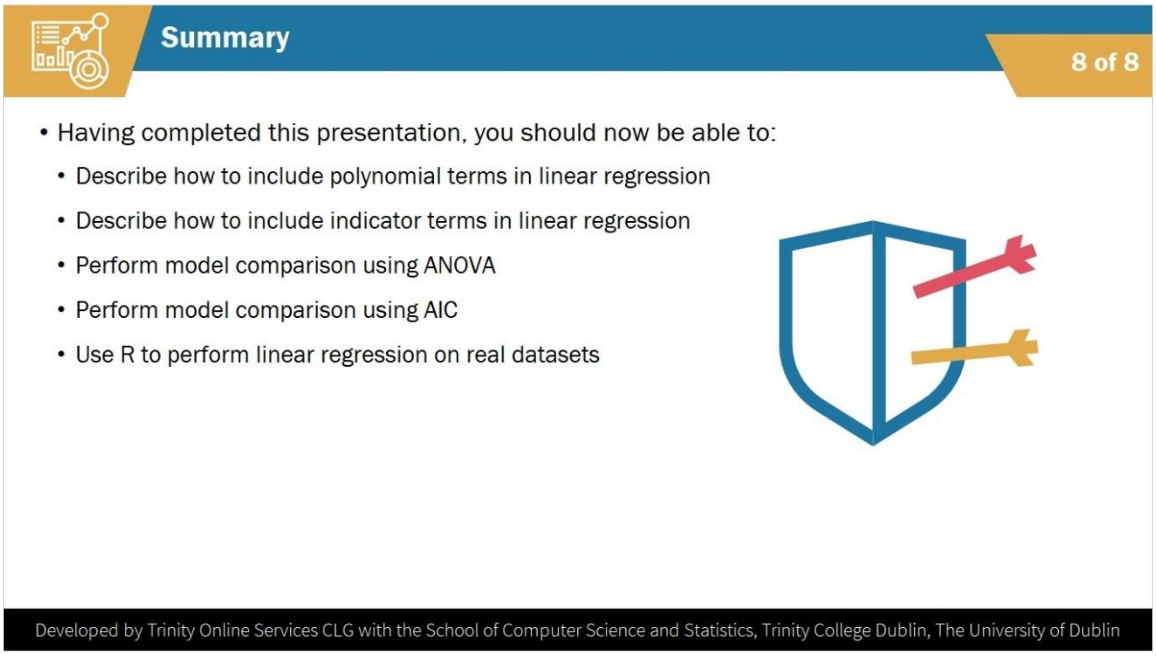
Tab 1.6: R: Function Step



The function ``step'' we used before is an iterative procedure which tries all the possible models (in a greedy way for computational reasons) and selects the model with the smallest AIC.

Note that given m covariate, there are 2^m possible linear regression models we can build by selecting subsets of covariates. For m=20 covariates, there are therefore more than 1 million possible linear regression models. The function Step from R uses a greedy selection approach that avoids to run all the 2^m models to compute their respective AIC.

Slide 8: Summary



Having completed this presentation, you should now be able to:

Describe how to include polynomial terms in linear regression,

Describe how to include indicator terms in linear regression,

Perform model comparison using ANOVA,

Perform model comparison using AIC, and

Use R to perform linear regression on real datasets.