

tidy models

Freddy Hernandez

tidymodels

tidymodels.org

PACKAGES GET STARTED LEARN HELP CONTRIBUTE ABOUT FIND

TIDYMODELS

The tidymodels framework is a collection of packages for modeling and machine learning using **tidyverse** principles.

Install tidymodels with:

```
install.packages("tidymodels")
```

tidymodels

rsample

parsnip

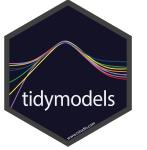
TUNE

recipes

yardstick

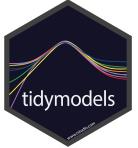
Max Kuhn – Posit





¿Cómo se instala tidy models?

```
1 install.packages("tidymodels")
```



¿Cómo se carga tidymodels?

```
1 library(tidymodels)
2 #> — Attaching packages —————— tidymodels 1.1.1 —
3 #> ✓ broom      1.0.5    ✓ rsample     1.2.0
4 #> ✓ dials      1.2.0    ✓ tibble      3.2.1
5 #> ✓ dplyr      1.1.2    ✓ tidyverse   1.3.0
6 #> ✓ infer      1.0.5    ✓ tune        1.1.2
7 #> ✓ modeldata   1.2.0    ✓ workflows   1.1.3
8 #> ✓ parsnip     1.1.1    ✓ workflowsets 1.0.1
9 #> ✓ purrr       1.0.2    ✓ yardstick   1.2.0
10 #> ✓ recipes     1.0.8
11 #> — Conflicts —————— tidymodels_conflicts() —
12 #> ✗ purrr::discard() masks scales::discard()
13 #> ✗ dplyr::filter()  masks stats::filter()
14 #> ✗ dplyr::lag()    masks stats::lag()
15 #> ✗ recipes::step() masks stats::step()
16 #> • Learn how to get started at https://www.tidymodels.org/start/
```

Su turno



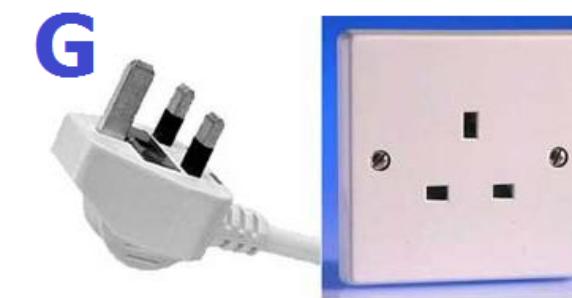
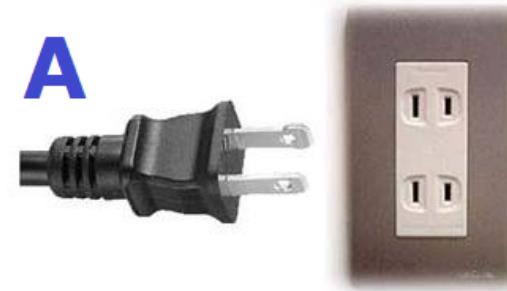
Instale y cargue tidymodels así:

```
1 install.packages("tidymodels")
2 library(tidymodels)
```

¿Qué debemos conocer para este tutorial?

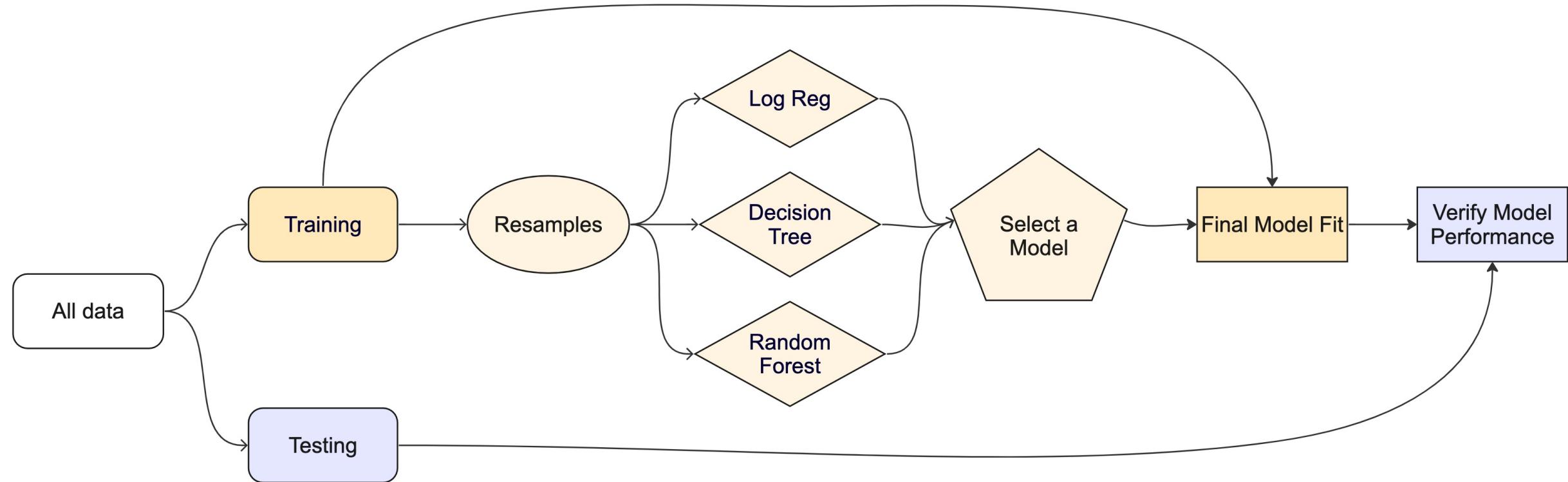
- Haber usado el operador `%>%` de magrittr o el operador `|>` de R base.
- Estar familiarizado/a con las funciones de dplyr, tidyr y ggplot2.
- Tener conceptos estadísticos básicos.
- **No** necesitas tener familiaridad intermedia o experta con modelado o aprendizaje automático.

¿Para qué sirve tidy models?



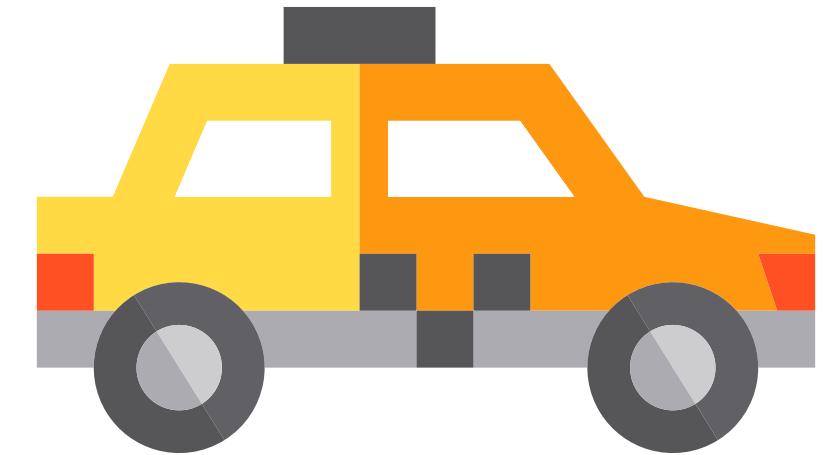
Adaptador

El panorama completo al modelar



Datos sobre viajes en taxi en Chicago

- La ciudad de Chicago publica datos anonimizados a nivel de viaje sobre los trayectos en taxi en la ciudad.
- Hemos extraído una muestra de 10000 viajes que tuvieron lugar a principios de 2022.
- Escribe [?taxi](#) para obtener más información sobre este conjunto de datos, incluyendo referencias.



Datos sobre viajes en taxi en Chicago

- $n = 10000$
- La variable de interés es `tip`, con niveles "yes" y "no"
- **nominales:** company, local, dow, month.
- **numericas:** distance, hour.



Los datos

```
1 library(tidymodels)
2
3 taxi
4 #> # A tibble: 10,000 × 7
5 #>   tip     distance company
6 #>   <dbl>    <dbl> <fct>
7 #> 1 17.2     1.47 Chicago Independents
8 #> 2 0.88     1.47 City Service
9 #> 3 18.1     1.47 other
10 #> 4 20.7     1.47 Chicago Independents
11 #> 5 12.2     1.47 Chicago Independents
12 #> 6 0.94     1.47 Sun Taxi
13 #> 7 17.5     1.47 Flash Cab
14 #> 8 17.7     1.47 other
15 #> 9 1.85     1.47 Taxicab Insurance Agency Llc
16 #> 10 1.47    1.47 City Service
17 #> # i 9,990 more rows
```

	tip	distance	company	local	dow	month	hour
	<dbl>	<dbl>	<fct>	<fct>	<fct>	<fct>	<int>
1	17.2	1.47	Chicago Independents	no	Thu	Feb	16
2	0.88	1.47	City Service	yes	Thu	Mar	8
3	18.1	1.47	other	no	Mon	Feb	18
4	20.7	1.47	Chicago Independents	no	Mon	Apr	8
5	12.2	1.47	Chicago Independents	no	Sun	Mar	21
6	0.94	1.47	Sun Taxi	yes	Sat	Apr	23
7	17.5	1.47	Flash Cab	no	Fri	Mar	12
8	17.7	1.47	other	no	Sun	Jan	6
9	1.85	1.47	Taxicab Insurance Agency Llc	no	Fri	Apr	12
10	1.47	1.47	City Service	no	Tue	Mar	14

Su turno



Obtenga los registros 320, 558, 784 y 5600 de taxi. El resultado debe ser este:

```
#> # A tibble: 4 × 7
#>   tip     distance company           local   dow month   hour
#>   <dbl>     <dbl> <fct>          <fct>  <fct> <fct> <int>
#> 1 17.3      17.3  other            no     Fri  Jan     16
#> 2 18.1      18.1  Flash Cab       no     Mon  Jan     20
#> 3 17.3      17.3  Flash Cab       no     Tue  Mar      9
#> 4 0         0     Taxicab Insurance Agency Llc yes   Wed  Feb     11
```

Solución

```
1 taxi |> slice(c(320, 558, 784, 5600))
2 #> # A tibble: 4 × 7
3 #>   tip    distance company      local  dow month hour
4 #>   <dbl>     <dbl> <fct>      <fct> <fct> <fct> <int>
5 #> 1 17.3     17.3  other       no     Fri  Jan    16
6 #> 2 18.1     18.1  Flash Cab  no     Mon  Jan    20
7 #> 3 17.3     17.3  Flash Cab  no     Tue  Mar     9
8 #> 4 0        0     Taxicab Insurance Agency Llc yes   Wed  Feb    11
```

Explorando la variable respuesta

Vamos a construir una tabla de frecuencia relativa para la variable respuesta.

```
1 taxi |>
2   select(tip) |>
3   table() |>
4   prop.table()
5 #> tip
6 #>     yes      no
7 #> 0.9209 0.0791
```

División de los datos

En machine learning se deben dividir los datos en al menos dos grupos:

- **training set**: se usa para estimar los parámetros del modelo (entrenamiento).
- **test set**: se usa para medir el desempeño del modelo.

No use  el test set en el entrenamiento.

División de los datos



Funciones para dividir los datos

- `initial_split(data, prop = 3/4, strata = NULL, ...)`: sirve para crear la partición.
- `training()`: sirve para extraer training de la partición.
- `testing()`: sirve para extraer test de la partición.



Creando la partición

```
1 set.seed(123)
2 taxi_split <- initial_split(data=taxi)
3
4 taxi_split
5 #> <Training/Testing/Total>
6 #> <7500/2500/10000>
```

Obteniendo los dos conjuntos

```
1 taxi_train <- taxi_split |> training()  
2  
3 taxi_test  <- taxi_split |> testing()
```

El conjunto de train

```
1 taxi_train
2 #> # A tibble: 7,500 × 7
3 #>   tip     distance company           local  dow month hour
4 #>   <dbl>    <dbl> <fct>          <fct> <fct> <fct> <int>
5 #> 1 0.7     Taxi Affiliation Services yes   Tue  Mar   18
6 #> 2 0.99    Sun Taxi                   yes   Tue  Jan    8
7 #> 3 1.78    other                     no    Sat  Mar   22
8 #> 4 0       Taxi Affiliation Services yes   Wed  Apr   15
9 #> 5 0       Taxi Affiliation Services no    Sun  Jan   21
10 #> 6 2.3     other                     no    Sat  Apr   21
11 #> 7 6.35    Sun Taxi                   no    Wed  Mar   16
12 #> 8 2.79    other                     no    Sun  Feb   14
13 #> 9 16.6    other                     no    Sun  Apr   18
14 #> 10 0.02   Chicago Independents yes   Sun  Apr   15
15 #> # i 7,490 more rows
```

El conjunto de test

```
1 taxi_test
2 #> # A tibble: 2,500 × 7
3 #>   tip     distance company
4 #>   <dbl>    <dbl> <fct>
5 #> 1 0.94    0.94 Sun Taxi
6 #> 2 1        1     Taxi Affiliation Services
7 #> 3 1.91    1.91 Flash Cab
8 #> 4 1.1      1.1   Chicago Independents
9 #> 5 11.7     11.7  other
10 #> 6 17.8     17.8  City Service
11 #> 7 0.53    0.53 Taxicab Insurance Agency Llc
12 #> 8 1.14    1.14  City Service
13 #> 9 1.77    1.77  other
14 #> 10 18.6    18.6  Flash Cab
15 #> # i 2,490 more rows
```

	tip	distance	company	local	dow	month	hour
	<dbl>	<dbl>	<fct>	<fct>	<fct>	<fct>	<int>
1	0.94	0.94	Sun Taxi	yes	Sat	Apr	23
2	1	1	Taxi Affiliation Services	no	Mon	Feb	18
3	1.91	1.91	Flash Cab	no	Wed	Apr	15
4	1.1	1.1	Chicago Independents	no	Sat	Mar	10
5	11.7	11.7	other	no	Wed	Mar	18
6	17.8	17.8	City Service	no	Mon	Mar	9
7	0.53	0.53	Taxicab Insurance Agency Llc	yes	Wed	Apr	8
8	1.14	1.14	City Service	no	Wed	Mar	14
9	1.77	1.77	other	no	Thu	Apr	15
10	18.6	18.6	Flash Cab	no	Thu	Apr	12
11	# i	2,490	more rows				

Su turno



- Divida los datos en 80% para train y 20% para test.
- Coloque dentro de `initial_split()` el argumento `strata=tip` para que haga la división teniendo en cuenta la variable `tip`.
- Use la semilla 123 dentro de `set.seed()`.
- Llame a los conjuntos `taxi_train` y `taxi_test`.

Solución

```
1 set.seed(123)
2 taxi_split <- initial_split(data=taxi, prop=0.8, strata=tip)
3 taxi_train <- taxi_split |> training()
4 taxi_test  <- taxi_split |> testing()
```

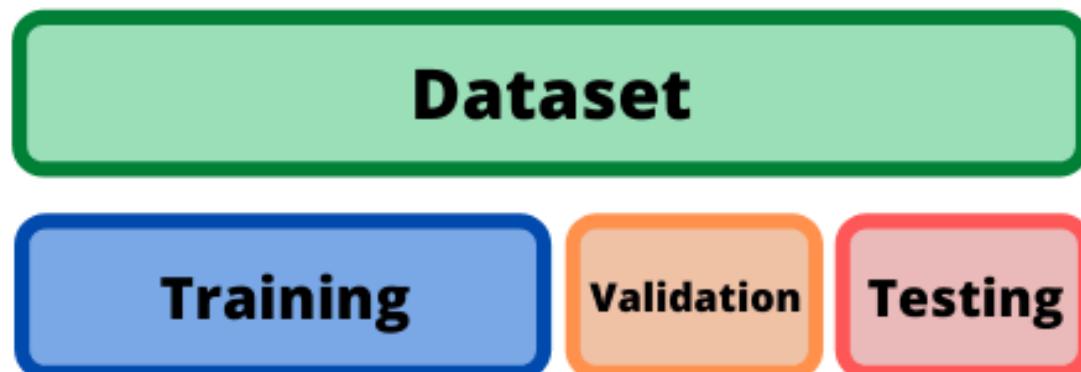
El nuevo taxi_train

```
1 taxi_train
2 #> # A tibble: 8,000 × 7
3 #>   tip    distance company
4 #>   <dbl>     <dbl> <fct>
5 #> 1 17.2      1.41 Chicago Independents
6 #> 2 0.88      0.789 City Service
7 #> 3 18.1      2.14 other
8 #> 4 12.2      1.45 Chicago Independents
9 #> 5 0.94      0.782 Sun Taxi
10 #> 6 17.5      2.05 Flash Cab
11 #> 7 17.7      2.01 other
12 #> 8 1.85      2.01 Taxicab Insurance Agency Llc
13 #> 9 0.53      1.58 Sun Taxi
14 #> 10 6.65     1.82 Taxicab Insurance Agency Llc
15 #> # i 7,990 more rows
```

	tip	distance	company	local	dow	month	hour
	<dbl>	<dbl>	<fct>	<fct>	<fct>	<fct>	<int>
1	17.2	1.41	Chicago Independents	no	Thu	Feb	16
2	0.88	0.789	City Service	yes	Thu	Mar	8
3	18.1	2.14	other	no	Mon	Feb	18
4	12.2	1.45	Chicago Independents	no	Sun	Mar	21
5	0.94	0.782	Sun Taxi	yes	Sat	Apr	23
6	17.5	2.05	Flash Cab	no	Fri	Mar	12
7	17.7	2.01	other	no	Sun	Jan	6
8	1.85	2.01	Taxicab Insurance Agency Llc	no	Fri	Apr	12
9	0.53	1.58	Sun Taxi	no	Tue	Mar	18
10	6.65	1.82	Taxicab Insurance Agency Llc	no	Sun	Apr	11
11	# i	7,990	more rows				

Funciones para dividir los datos

- `initial_validation_split(data, prop = c(0.6, 0.2), strata = NULL, ...)`: sirve para crear la partición.
- `training()`: sirve para extraer training de la partición.
- `testing()`: sirve para extraer test de la partición.
- `validation()`: sirve para extraer validation de la partición.

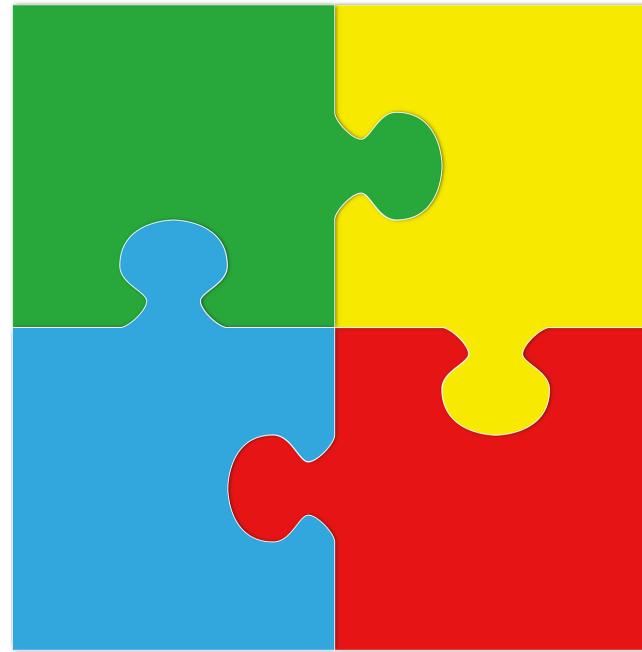


Modelos a usar en este tutorial

- Árbol de clasificación.
- Regresión logística.

Partes de un tidymodel

1. Especificación del modelo.
2. La receta.
3. El workflow.
4. El entrenamiento.





Elementos para especificar el modelo

- Elegir el modelo (model).
- Elegir el motor (engine).
- Definir el modo (regresión o clasificación).

Especificando el árbol

- Elegir el modelo (model).
- Elegir el motor (engine).
- Definir el modo (regresión o clasificación).

```
1 tree_spec <- decision_tree() |>
2   set_args(cost_complexity = 0.0001) |>
3   set_engine("rpart") |>
4   set_mode("classification")
5
6 tree_spec
7 #> Decision Tree Model Specification (classification)
8 #>
9 #> Main Arguments:
10 #>   cost_complexity = 1e-04
11 #>
12 #> Computational engine: rpart
```

Especificando la regresión logística

- Elegir el modelo (model).
- Elegir el motor (engine).
- Definir el modo (regresión o clasificación).

```
1 lr_spec <- logistic_reg() |>
2   set_args(family = binomial(link="logit")) |>
3   set_engine("glm") |>
4   set_mode("classification")
5
6 lr_spec
7 #> Logistic Regression Model Specification (classification)
8 #>
9 #> Computational engine: glm
```

Lista de todos los modelos

<https://www.tidymodels.org/find/parsnip/>

Search parsnip models

Find model types, engines, and arguments to fit and predict in the tidymodels framework.

To learn about the parsnip package, see [Get Started: Build a Model](#). Use the tables below to find [model types](#) and [engines](#).

Show 25 entries

Search:

title	model	engine	topic	mode	package
All	All	All	All	All	All
C5.0 rule-based classification models	C5_rules	C5.0	C5_rules_C5.0	classification	rules
ADAM Regression Models	adam_reg	adam	adam_reg	regression	modeltimer
ADAM Regression Models	adam_reg	auto_adam	adam_reg	regression	modeltimer



Creando la receta

El paquete `recipes` tiene muchas funciones que nos permite transformar las variables. Consultar <https://recipes.tidymodels.org>.

```
1 general_rec <- recipe(tip ~ ., data=taxi)
```



Creando el workflow

Para crear el workflow se usa la función `workflow()`.

```
1 tree_wf <- workflow() |>
2   add_model(tree_spec) |>
3   add_recipe(general_rec)
4
5 lr_wf <- workflow() |>
6   add_model(lr_spec) |>
7   add_recipe(general_rec)
```

Entrenando el modelo

Para entrenar el modelo se usa la función `fit()`.

```
1 # The tree
2 tree_fit <- tree_wf |>
3   fit(data = taxi_train)
4
5 # The LR
6 lr_fit <- lr_wf |>
7   fit(data = taxi_train)
```

Haciendo predicciones

Vamos a predecir la variable `tip` usando `taxi_train` y el árbol de clasificación por medio de la función `predict(object, new_data, type=c("numeric", "class", "prob", ...))`.

```
1 predict(object=tree_fit, new_data=taxi_train)
2 #> # A tibble: 8,000 × 1
3 #>   .pred_class
4 #>   <fct>
5 #>   1 yes
6 #>   2 yes
7 #>   3 yes
8 #>   4 yes
9 #>   5 yes
10 #>  6 yes
11 #>  7 yes
12 #>  8 yes
13 #>  9 yes
14 #> 10 yes
15 #> # i 7,990 more rows
```

Agregando las predicciones a los datos

Es posible agregar los resultados del modelo a los datos.

```
1 augment(x=tree_fit, new_data = taxi_train)
2 #> # A tibble: 8,000 × 10
3 #>   tip    distance company local dow month hour .pred_class .pred_yes .pred_no
4 #>   <dbl>     <dbl> <fct>   <fct> <fct> <fct> <int> <fct>      <dbl>    <dbl>
5 #> 1 yes     17.2 Chicag... no     Thu   Feb     16 yes        0.967  0.0333
6 #> 2 yes      0.88 City S... yes   Thu   Mar      8 yes        0.935  0.0646
7 #> 3 yes     18.1 other    no     Mon   Feb     18 yes        0.967  0.0333
8 #> 4 yes     12.2 Chicag... no     Sun   Mar     21 yes        0.949  0.0507
9 #> 5 yes      0.94 Sun Ta... yes   Sat   Apr     23 yes        0.821  0.179
10 #> 6 yes     17.5 Flash ... no    Fri   Mar     12 yes        0.967  0.0333
11 #> 7 yes     17.7 other    no     Sun   Jan      6 yes        0.967  0.0333
12 #> 8 yes      1.85 Taxica... no    Fri   Apr     12 yes        0.938  0.0616
13 #> 9 yes      0.53 Sun Ta... no    Tue   Mar     18 yes        0.938  0.0616
14 #> 10 yes     6.65 Taxica... no   Sun   Apr     11 yes        0.931  0.0694
15 #> # i 7,990 more rows
```

Su turno



Repita el procedimiento anterior pero usando `taxi_test`. El resultado debe ser el siguiente:

```
#> # A tibble: 2,000 × 10
#>   tip     distance company local dow month hour .pred_class .pred_yes .pred_no
#>   <fct>    <dbl> <fct>    <fct> <fct> <fct> <int> <fct>           <dbl>    <dbl>
#> 1 yes      20.7 Chicag... no     Mon   Apr      8 yes            0.967  0.0333
#> 2 yes       1.47 City S... no    Tue   Mar     14 yes            0.938  0.0616
#> 3 yes        1   Taxi A... no    Mon   Feb     18 yes            0.938  0.0616
#> 4 yes       1.91 Flash ... no   Wed   Apr     15 yes            0.948  0.0519
#> 5 yes       17.2 City S... no   Mon   Apr      9 yes            0.967  0.0333
#> 6 yes       17.8 City S... no   Mon   Mar      9 yes            0.967  0.0333
#> 7 yes       0.53 Taxica... yes   Wed   Apr      8 yes            0.935  0.0646
#> 8 yes       1.77 other    no    Thu   Apr     15 yes            0.938  0.0616
#> 9 yes       18.6 Flash ... no   Thu   Apr     12 yes            0.967  0.0333
#> 10 no       1.13 other   no    Sat   Feb     14 yes            0.938  0.0616
#> # i 1,990 more rows
```

Solución

```
1 augment(x=tree_fit, new_data = taxi_test)
```

Matriz de confusión

		Observed Values	
		Tipping - yes	Tipping - no
Predicted Values	Tipping - yes	TP	FP
	Tipping - no	FN	TN



Creando la matriz de confusión

La matriz de confusión se obtiene con `conf_mat()` aplicada luego de `augment()`.

```
1 augment(x=tree_fit, new_data = taxi_train) %>%
2   conf_mat(truth = tip, estimate = .pred_class)
3 #>           Truth
4 #> Prediction  yes    no
5 #>           yes 7341  536
6 #>           no   43   80
```

El paquete `yardstick` tiene funciones útiles que se pueden consultar en <https://yardstick.tidymodels.org>.

Su turno



Construya la matriz de confusión pero usando `taxi_test`. El resultado debe ser el siguiente:

```
#>           Truth  
#> Prediction  yes    no  
#>          yes 1807   165  
#>          no    18    10
```

Solución

```
1 augment(x=tree_fit, new_data = taxi_test) %>%
2   conf_mat(truth = tip, estimate = .pred_class)
```

Precisión o accuracy

		Observed Values	
		Tipping - yes	Tipping - no
Predicted Values	Tipping - yes	TP	FP
	Tipping - no	FN	TN

Accuracy

$$\frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}$$



Calculando la precisión

```
1 augment(x=tree_fit, new_data = taxi_train) %>%
2   accuracy(truth = tip, estimate = .pred_class)
3 #> # A tibble: 1 × 3
4 #>   .metric   .estimator .estimate
5 #>   <chr>     <chr>          <dbl>
6 #> 1 accuracy  binary         0.928
```

Su turno



Calcule la precisión pero usando `taxi_test`. El resultado debe ser el siguiente:

```
#> # A tibble: 1 × 3  
#>   .metric  .estimator .estimate  
#>   <chr>    <chr>        <dbl>  
#> 1 accuracy binary     0.908
```

Solución

```
1 augment(x=tree_fit, new_data = taxi_test) %>%
2   accuracy(truth = tip, estimate = .pred_class)
```

Recursos útiles en la web

Les recomiendo consultar el siguiente material para conocer más sobre tidymodels.

- Curso completo: <https://workshops.tidymodels.org>
- La organización: <https://www.tidymodels.org>
- Libro: Tidy Modeling with R <https://www.tmwr.org>

GRACIAS!

