```
# Import libraries. You may or may not use all of these.
!pip install -q git+https://github.com/tensorflow/docs
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd

try:
  # %tensorflow_version only exists in Colab.
  %tensorflow_version 2.x
except Exception:
  pass
import tensorflow as tf

from tensorflow import keras
from tensorflow.keras import layers

import tensorflow_docs as tfdocs
import tensorflow_docs.plots
import tensorflow_docs.modeling
```

```
  Preparing metadata (setup.py) ... done
Colab only includes TensorFlow 2.x; %tensorflow_version has no effect.
```

```
# Import data
!wget https://cdn.freecodecamp.org/project-data/health-costs/insurance.csv
dataset = pd.read_csv('insurance.csv')
dataset.tail()
```

```
--2025-04-03 21:22:43--  https://cdn.freecodecamp.org/project-data/health-costs/insurance.csv
Resolving cdn.freecodecamp.org (cdn.freecodecamp.org)... 172.67.70.149, 104.26.2.33, 104.26.3.33, ...
Connecting to cdn.freecodecamp.org (cdn.freecodecamp.org)|172.67.70.149|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 50264 (49K) [text/csv]
Saving to: ‘insurance.csv.4’

insurance.csv.4     100%[===================>]  49.09K  --.-KB/s    in 0.007s

2025-04-03 21:22:43 (6.53 MB/s) - ‘insurance.csv.4’ saved [50264/50264]
```

	age	sex	bmi	children	smoker	region	expenses
1333	50	male	31.0	3	no	northwest	10600.55
1334	18	female	31.9	0	no	northeast	2205.98
1335	18	female	36.9	0	no	southeast	1629.83
1336	21	female	25.8	0	no	southwest	2007.95
1337	61	female	29.1	0	yes	northwest	29141.36

Encoding

```
dataset['smoker'] = dataset['smoker'].map({'yes': 1, 'no': 0})
dataset['sex'] = dataset['sex'].map({'male': 1, 'female': 0})
dataset = pd.get_dummies(dataset, columns=['region'], prefix='region', dtype=int)
```

```
dataset.tail()
```

	age	sex	bmi	children	smoker	expenses	region_northeast	region_northwest	region_southeast	region_southwest
1333	50	1	31.0	3	0	10600.55	0	1	0	0
1334	18	0	31.9	0	0	2205.98	1	0	0	0
1335	18	0	36.9	0	0	1629.83	0	0	1	0
1336	21	0	25.8	0	0	2007.95	0	0	0	1
1337	61	0	29.1	0	1	29141.36	0	1	0	0

Split and normalization

```
from sklearn.model_selection import train_test_split

train_dataset, test_dataset = train_test_split(dataset, test_size=0.2, random_state=42)


train_labels = train_dataset.pop("expenses")
test_labels = test_dataset.pop("expenses")


normalizer = layers.Normalization(axis=-1)
normalizer.adapt(np.array(train_dataset))


print(normalizer.mean.numpy())
```

```
[[39.35701     0.5121495  30.56215     1.1074766   0.20560747  0.24953271
   0.23925234  0.26448599  0.24672897]]
```

Model

```
def build_model():
    model = keras.Sequential([
        normalizer,
        layers.Dense(64, activation='relu'),
        layers.Dense(64, activation='relu'),
        layers.Dense(1)
    ])

    model.compile(optimizer='adam', loss='mae', metrics=['mae', 'mse'])
    return model

model = build_model()
```

```
%%time
history = model.fit(
    train_dataset,
    train_labels,
    epochs=100,
    validation_split=0.2,
    verbose=0
)
```

```
CPU times: user 21.2 s, sys: 937 ms, total: 22.1 s
Wall time: 32 s
```

```
hist = pd.DataFrame(history.history)
hist['epoch'] = history.epoch
hist.tail()
```

	loss	mae	mse	val_loss	val_mae	val_mse	epoch
95	3154.497559	3154.497559	38680612.0	2923.582031	2923.582031	38427572.0	95
96	3144.361084	3144.361084	38548172.0	2912.873535	2912.873535	38149912.0	96
97	3132.996094	3132.996094	38338296.0	2897.994385	2897.994385	37831764.0	97
98	3123.119629	3123.119629	38134108.0	2888.388916	2888.388916	37677324.0	98
99	3112.208496	3112.208496	37905316.0	2876.478271	2876.478271	37428728.0	99

+ Code + Texte

Test

```
# RUN THIS CELL TO TEST YOUR MODEL. DO NOT MODIFY CONTENTS.
# Test model by checking how well the model generalizes using the test set.
loss, mae, mse = model.evaluate(test_dataset, test_labels, verbose=2)

print("Testing set Mean Abs Error: {:5.2f} expenses".format(mae))

if mae < 3500:
```

```
  print("You passed the challenge. Great job!")
else:
  print("The Mean Abs Error must be less than 3500. Keep trying.")

# Plot predictions.
test_predictions = model.predict(test_dataset).flatten()

a = plt.axes(aspect='equal')
plt.scatter(test_labels, test_predictions)
plt.xlabel('True values (expenses)')
plt.ylabel('Predictions (expenses)')
lims = [0, 50000]
plt.xlim(lims)
plt.ylim(lims)
_ = plt.plot(lims,lims)
```

```
9/9 - 0s - 6ms/step - loss: 2959.1465 - mae: 2959.1465 - mse: 34895536.0000
Testing set Mean Abs Error: 2959.15 expenses
You passed the challenge. Great job!
9/9 ━━━━━━━━━━━━━━━━━━━━ 0s 9ms/step
```