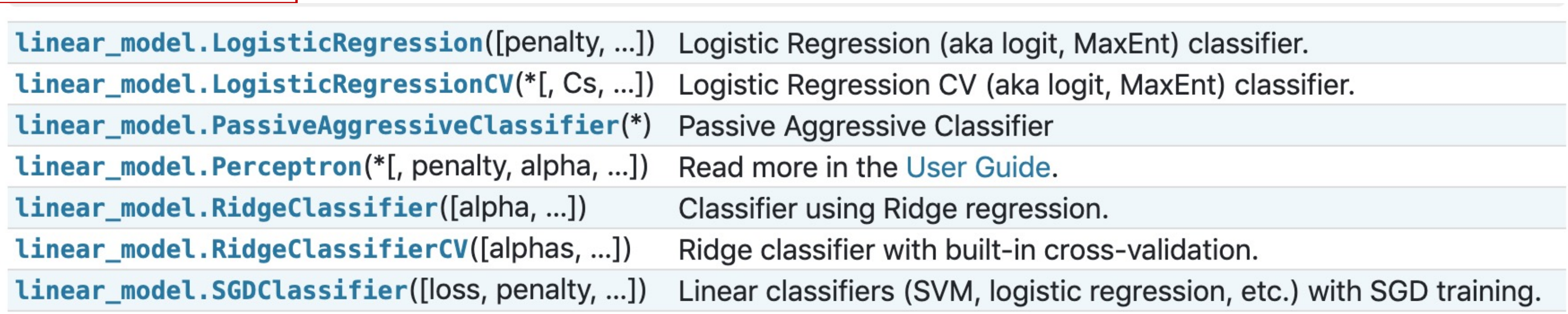
Applied Machine Learning Classification with Tensorflow

Ngan Le

thile@uark.edu

Recap: Classification with sklearn

- X_train, X_validate, y_train, y_validate = train_test_split(….)
- model = LogisticRegression(….)
- search = GridSearchC
- search.fit(X_train, y_
- print(search.best_est
- model = search.best_estimator_
- model.fit(X_train, y_train)
- predictions = model.predict(X_validate)



linear_model.LogisticRegression([penalty, …])	Logistic Regression (aka logit, MaxEnt) classifier.
linear_model.LogisticRegressionCV(*[, Cs, …])	Logistic Regression CV (aka logit, MaxEnt) classifier.
linear_model.PassiveAggressiveClassifier(*)	Passive Aggressive Classifier
linear_model.Perceptron(*[, penalty, alpha, …])	Read more in the User Guide.
linear_model.RidgeClassifier([alpha, …])	Classifier using Ridge regression.
linear_model.RidgeClassifierCV([alphas, …])	Ridge classifier with built-in cross-validation.
linear_model.SGDClassifier([loss, penalty, …])	Linear classifiers (SVM, logistic regression, etc.) with SGD training.

Use the **TensorFlow** toolkit to create a deep neural network that can perform **classification**

Dataset: UCI Heart Disease

Predicting the presence of heart disease

Dataset: Features

Feature	Description
age	age in years
sex	sex 0 = female 1 = male
cp	chest pain type 1 = typical angina 2 = atypical angina 3 = non-anginal pain 4 = asymptomatic

Dataset: Features (continued)

Feature	Description
trestbps	resting blood pressure in Hg
chol	serum cholesterol in mg/dl
fbs	is fasting blood sugar > 120 mg/dl 0 = false 1 = true
restecg	results of a resting electrocardiograph 0 = normal 1 = ST-T wave abnormality 2 = left ventricular hypertrophy

Dataset: Features (continued)

Feature	Description
thalach	max heart rate
exang	exercise induced angina 0 = no 1 = yes
oldpeak	measurement of an abnormal ST depression
slope	slope of peak of exercise ST segment 1 = upslope 2 = flat 3 = downslope

Dataset: Features (continued)

Feature	Description
ca	count of major blood vessels colored by fluoroscopy 0, 1, 2, 3, or 4
thal	presence heart condition 0 = unknown 1 = normal 2 = fixed defect 3 = reversible defect

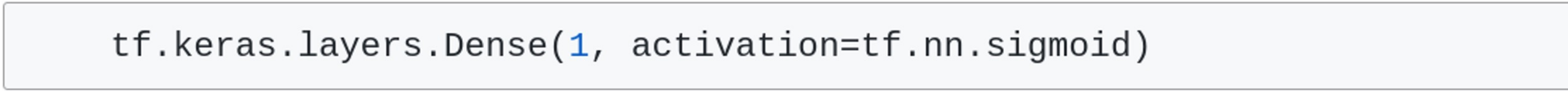
The values on the slides for some of these columns differ from the documentation.

For instance, the documentation for 'ca' states that the values range from 0-3, but there are 4s in the data. And the documentation for 'thal' says that the values are 3, 6, and 7, but the actual values in the data are 0, 1, 2, and 3.

Should always read the documentation, but you should also always look at the data and verify that the documentation is accurate

Classification with Tensorflow

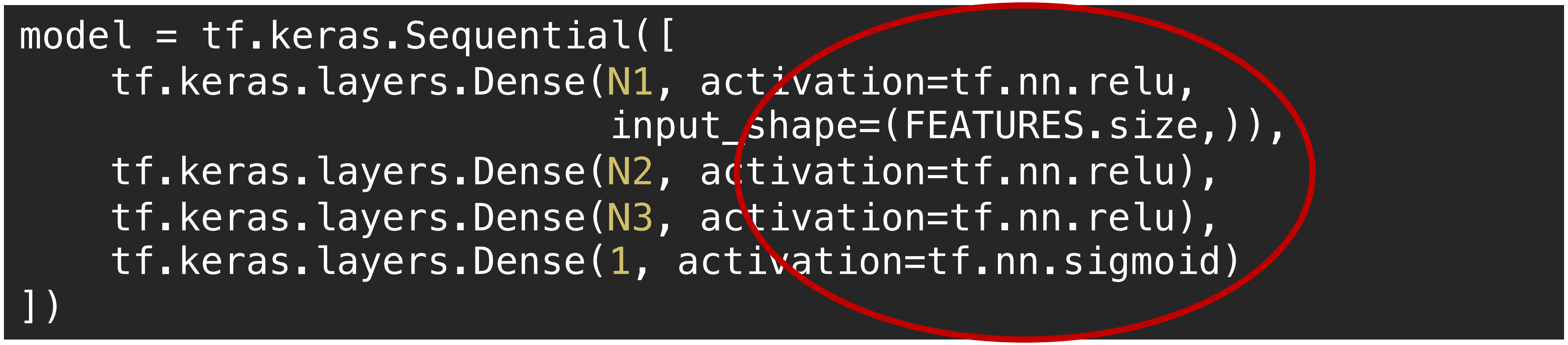
- X_train, X_validate, y_train, y_validate = train_test_split(….)



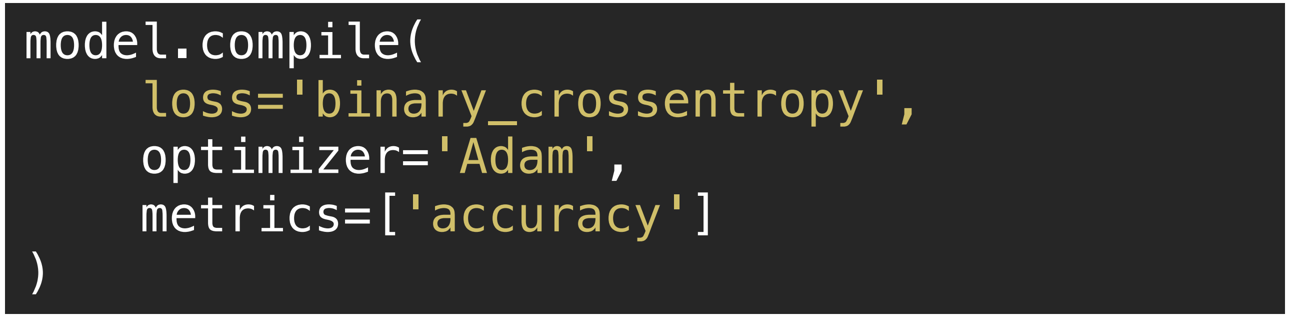
```
tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
```

Classification with Tensorflow

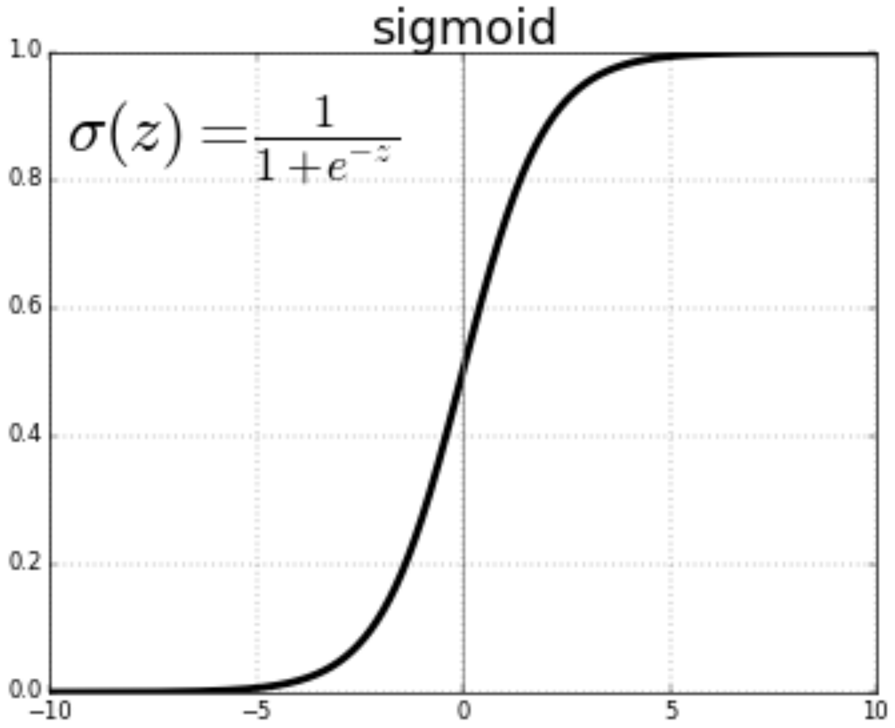
- X_train, X_validate, y_train, y_validate = train_test_split(….)

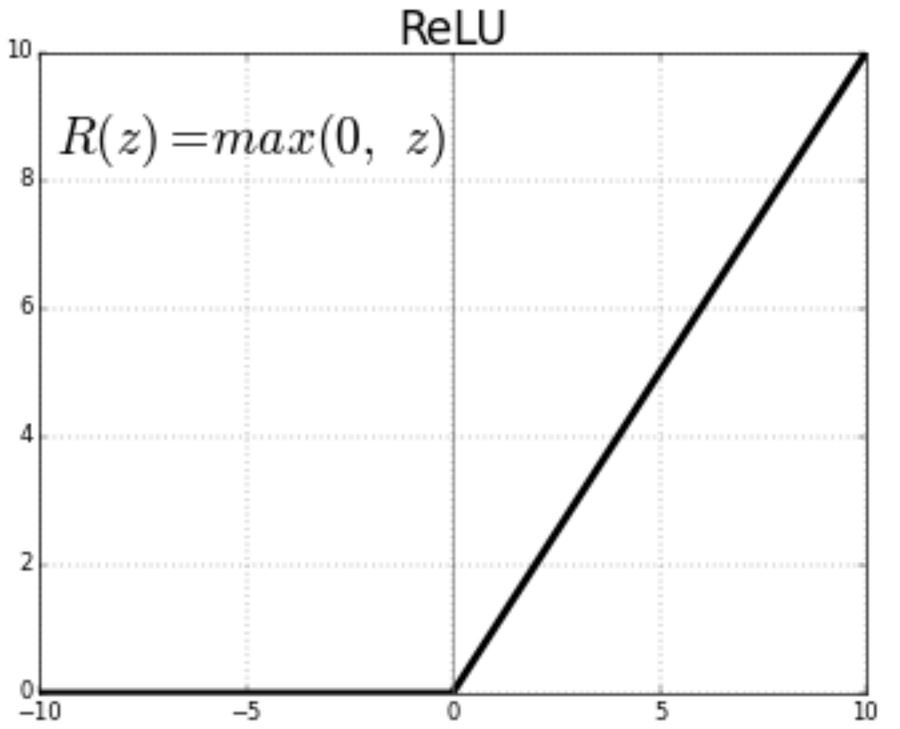
- 

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(N1, activation=tf.nn.relu,
                          input_shape=(FEATURES.size,)),
    tf.keras.layers.Dense(N2, activation=tf.nn.relu),
    tf.keras.layers.Dense(N3, activation=tf.nn.relu),
    tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
])
```



```
model.compile(
    loss='binary_crossentropy',
    optimizer='Adam',
    metrics=['accuracy']
)
```


sigmoid
$\sigma(z) = \frac{1}{1+e^{-z}}$
1.0
0.8
0.6
0.4
0.2
0.0
−10
−5
0
5
10


ReLU
$R(z) = max(0,\ z)$
10
8
6
4
2
0
−10
−5
0
5
10

Classification with Tensorflow

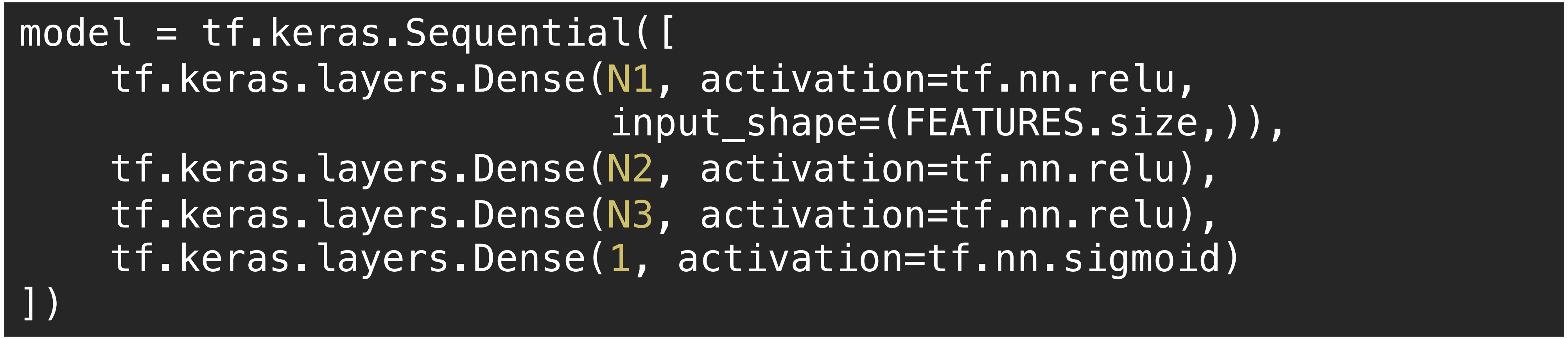
- X_train, X_validate, y_train, y_validate = train_test_split(….)
-
```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(N1, activation=tf.nn.relu,
                          input_shape=(FEATURES.size,)),
    tf.keras.layers.Dense(N2, activation=tf.nn.relu),
    tf.keras.layers.Dense(N3, activation=tf.nn.relu),
    tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
])
```

```
model.compile(
    loss='binary_crossentropy',
    optimizer='Adam',
    metrics=['accuracy']
)
```

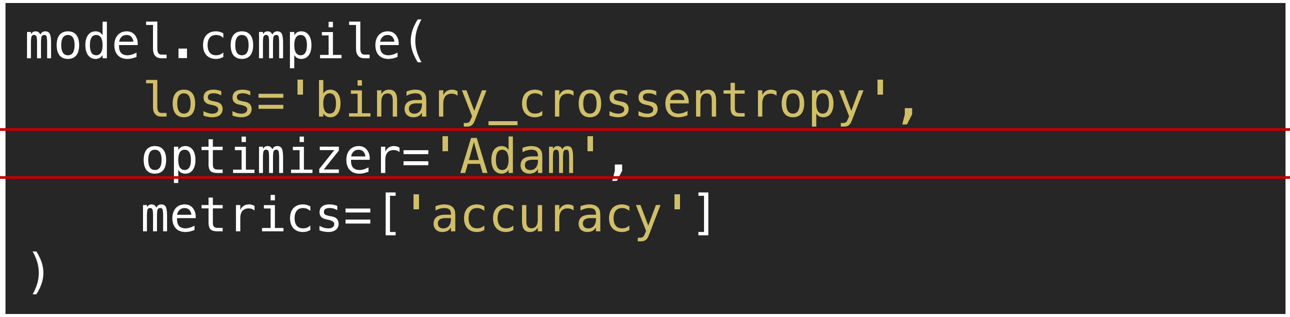
https://www.tensorflow.org/api_docs/python/tf/keras/losses

Classification with Tensorflow

- X_train, X_validate, y_train, y_validate = train_test_split(….)

- 

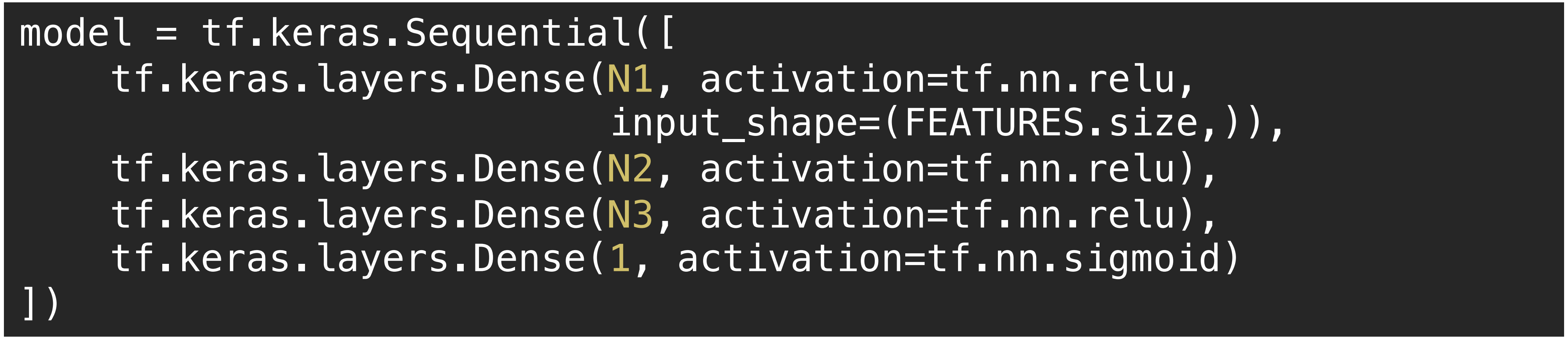
```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(N1, activation=tf.nn.relu,
                              input_shape=(FEATURES.size,)),
    tf.keras.layers.Dense(N2, activation=tf.nn.relu),
    tf.keras.layers.Dense(N3, activation=tf.nn.relu),
    tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
])
```



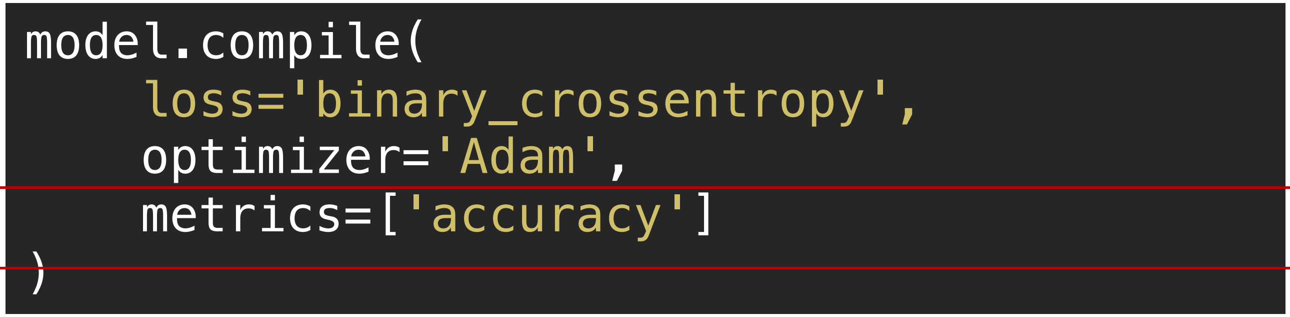
```
model.compile(
    loss='binary_crossentropy',
    optimizer='Adam',
    metrics=['accuracy']
)
```

https://www.tensorflow.org/api_docs/python/tf/keras/optimizers

Classification with Tensorflow

- X_train, X_validate, y_train, y_validate = train_test_split(….)
- 

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(N1, activation=tf.nn.relu,
                          input_shape=(FEATURES.size,)),
    tf.keras.layers.Dense(N2, activation=tf.nn.relu),
    tf.keras.layers.Dense(N3, activation=tf.nn.relu),
    tf.keras.layers.Dense(1, activation=tf.nn.sigmoid)
])
```



```
model.compile(
    loss='binary_crossentropy',
    optimizer='Adam',
    metrics=['accuracy']
)
```

https://www.tensorflow.org/api_docs/python/tf/keras/metrics

The Model: Early Stopping

```
tf.keras.callbacks.EarlyStopping(
    monitor='loss',
    min_delta=1e-3,
    patience=5,
)
```

Your Turn