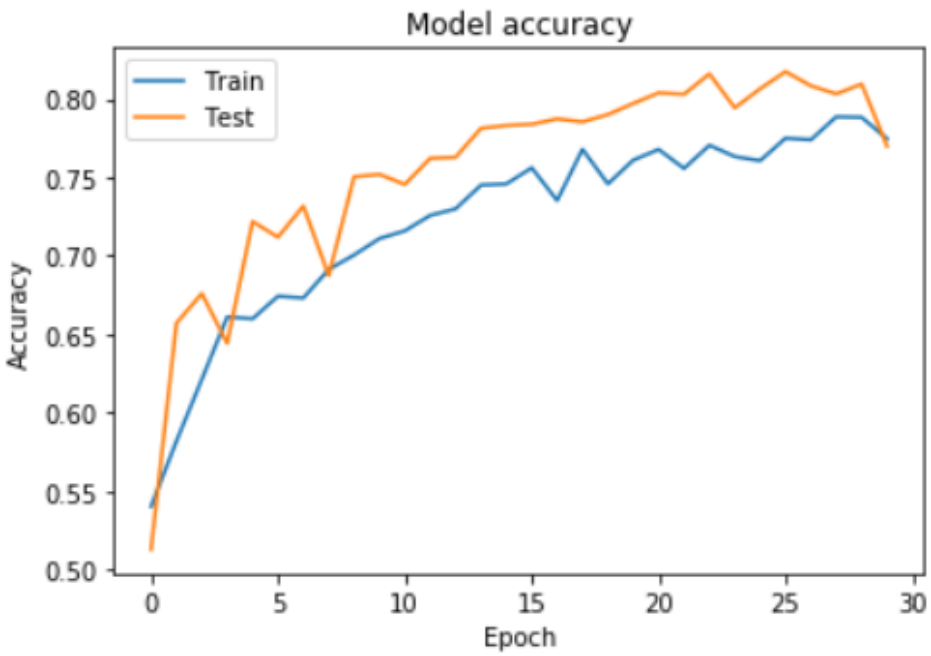
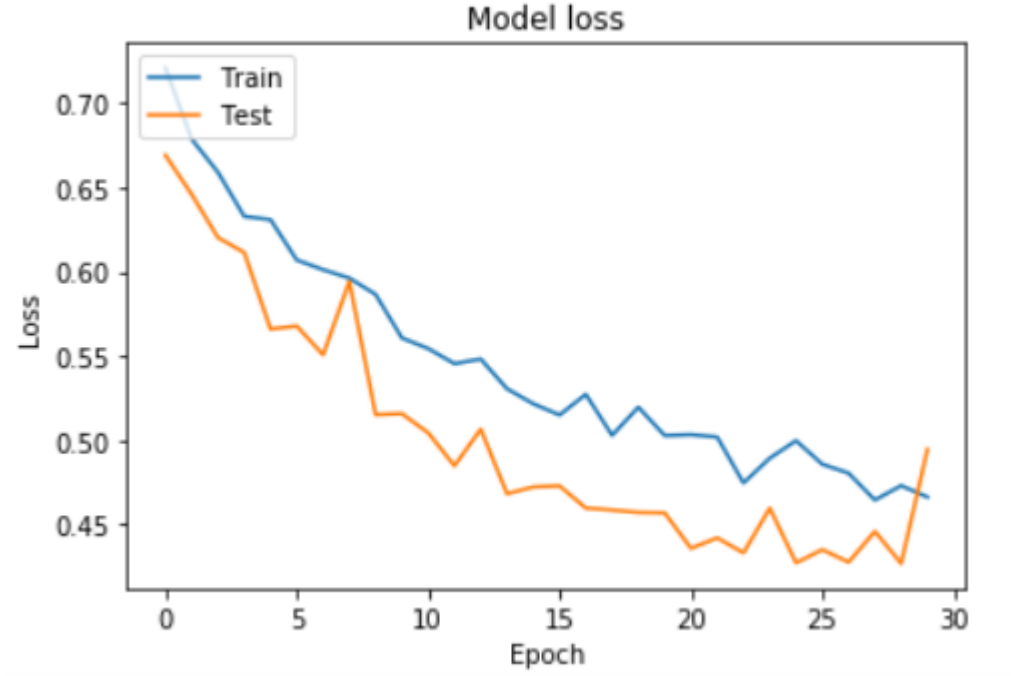
ALEXIS VALENCIA OSPINA – DATASET PERROS Y GATOS – 23.000 imágenes

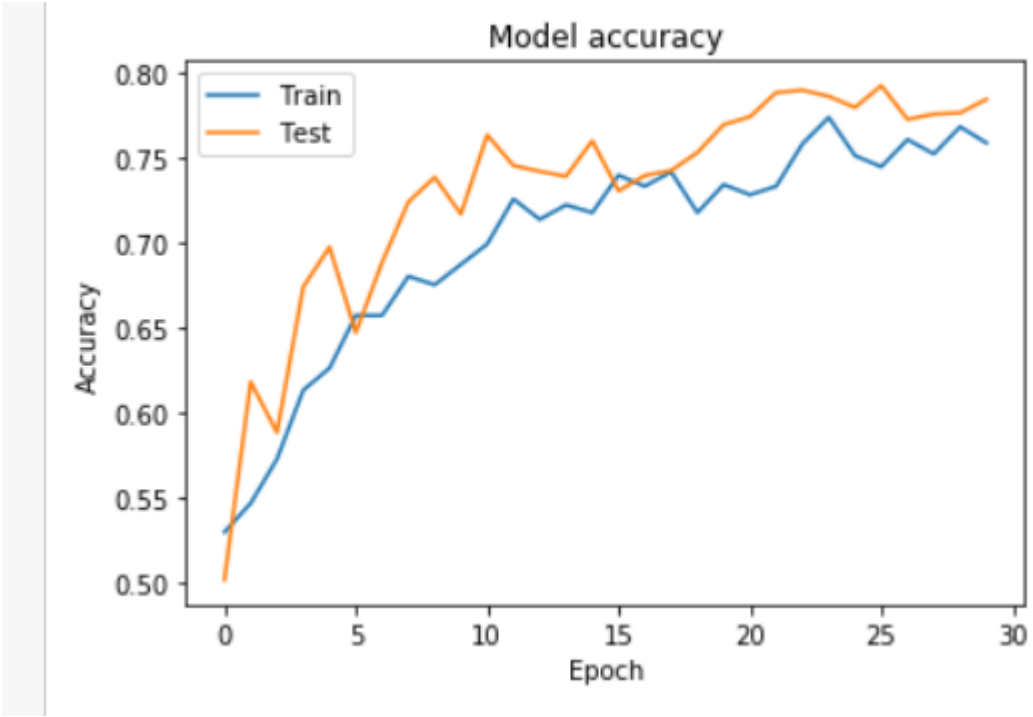
```
batch_size = 22

#C:\Users\alexi\Downloads\dogscats\train
train_data_dir = 'C:/Users/alexi/Downloads/dogscats/train'
validation_data_dir ='C:/Users/alexi/Downloads/dogscats/valid'

train_datagen = ImageDataGenerator(
        rescale=1./255,
        shear_range=0.2,
        zoom_range=0.2,
        horizontal_flip=True)

test_datagen = ImageDataGenerator(rescale=1./255)

train_generator = train_datagen.flow_from_directory(
        train_data_dir,
        target_size=(150, 150),
        batch_size=batch_size,
        class_mode='binary')

validation_generator = test_datagen.flow_from_directory(
        validation_data_dir,
        target_size=(150, 150),
        batch_size=32,
        class_mode='binary')
```

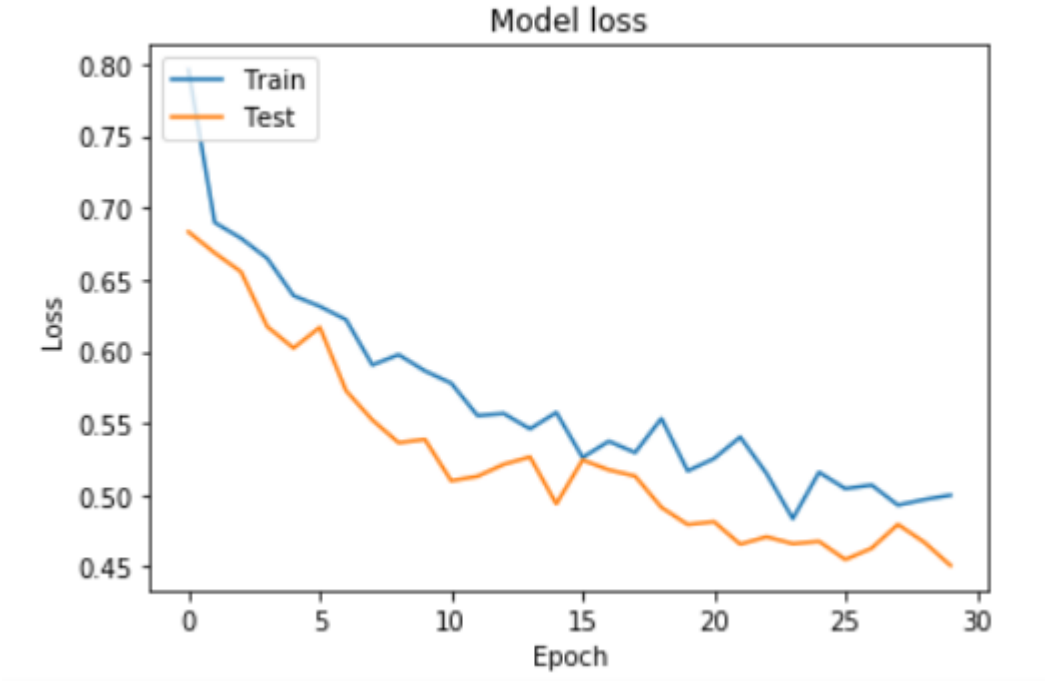
```
# MODEL -------------------------------------------------

model = Sequential()
model.add(Conv2D(32, kernel_size=(3, 3),
                 activation='relu',
                 input_shape=(150, 150, 3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

model.compile(loss=keras.losses.binary_crossentropy,
              optimizer=keras.optimizers.Adadelta(),
              metrics=['accuracy'])

# TRAINING ----------------------------------------------------

epochs = 30

h = model.fit_generator(
        train_generator,
        steps_per_epoch=120,
        epochs=epochs,
        validation_data=validation_generator,
        validation_steps=700
)
```


Model accuracy
Train
Test
Accuracy
0.80
0.75
0.70
0.65
0.60
0.55
0.50
0
5
10
15
20
25
30
Epoch

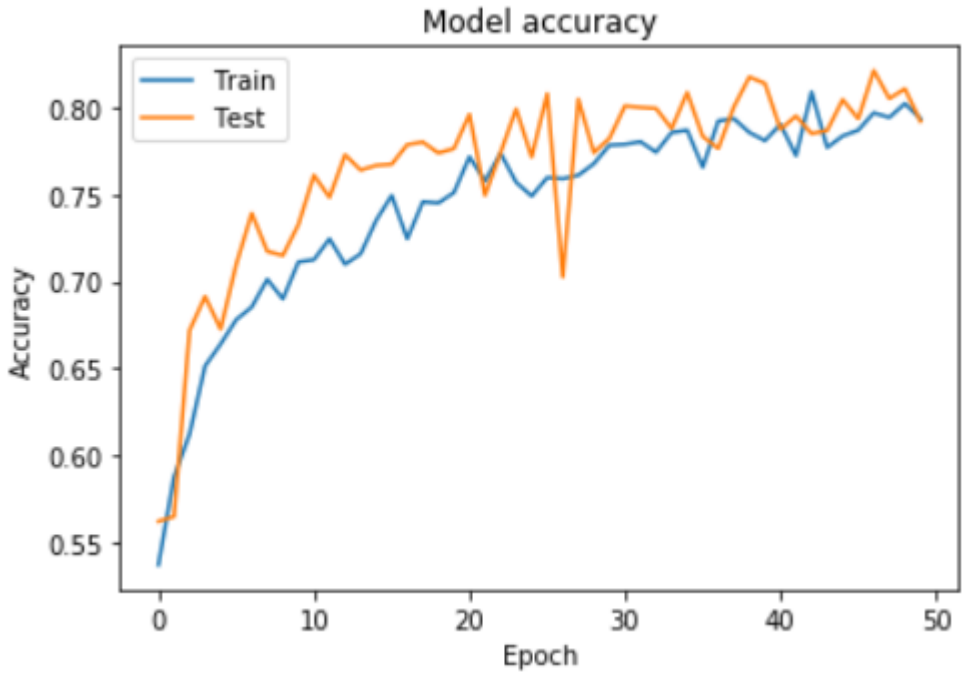

Model loss
Train
Test
Loss
0.70
0.65
0.60
0.55
0.50
0.45
0
5
10
15
20
25
30
Epoch

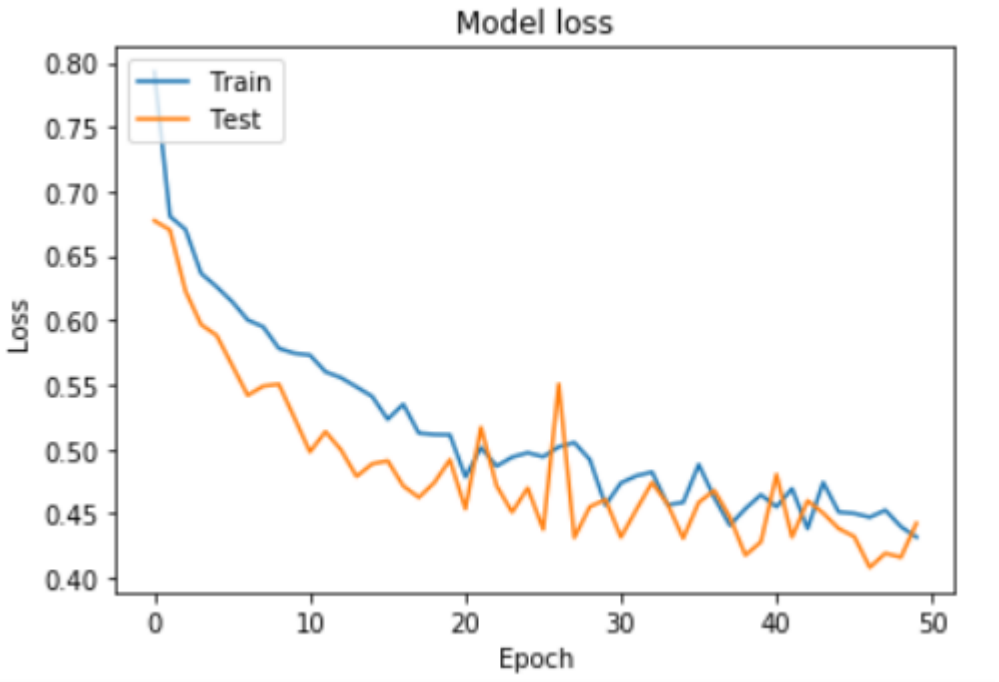
- Cambios para la siguiente prueba
  - Batch_size = 20
  - Capa oculta → 100 neuronas
  - Epochs = 30
  - Steps_per epoch = 100
  - Validation_steps = 800



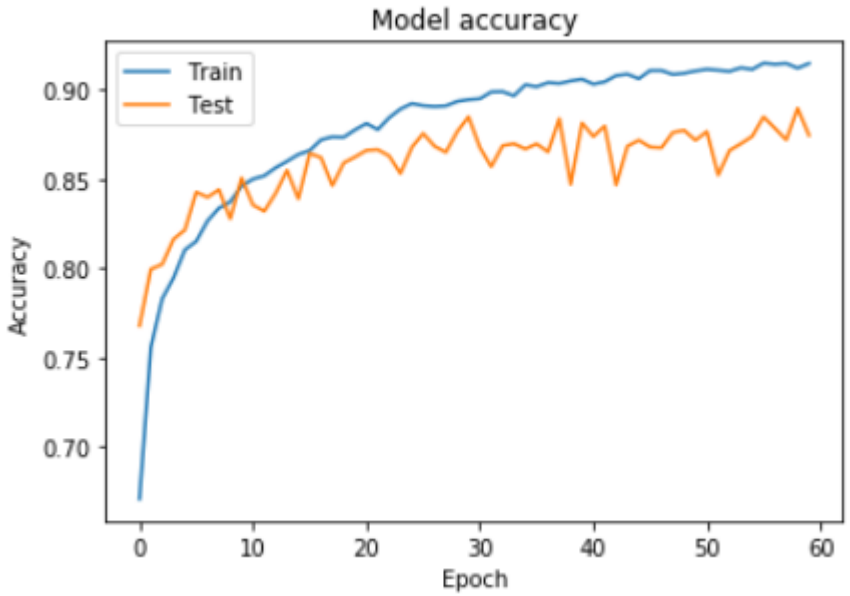


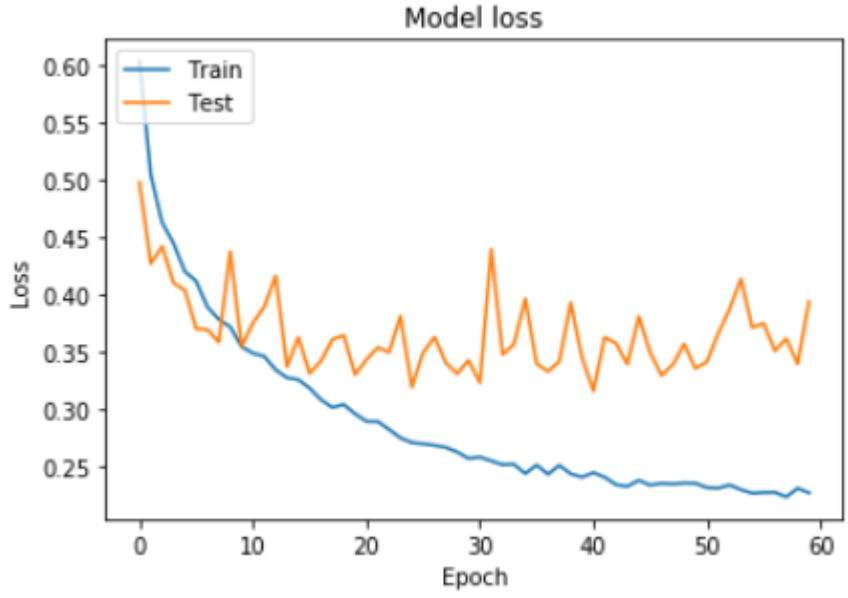
- Cambios para la siguiente prueba
  - Batch_size = 15
  - Capa oculta → 200 neuronas
  - Epochs = 50
  - Steps_per epoch = 100
  - Validation_steps = 600





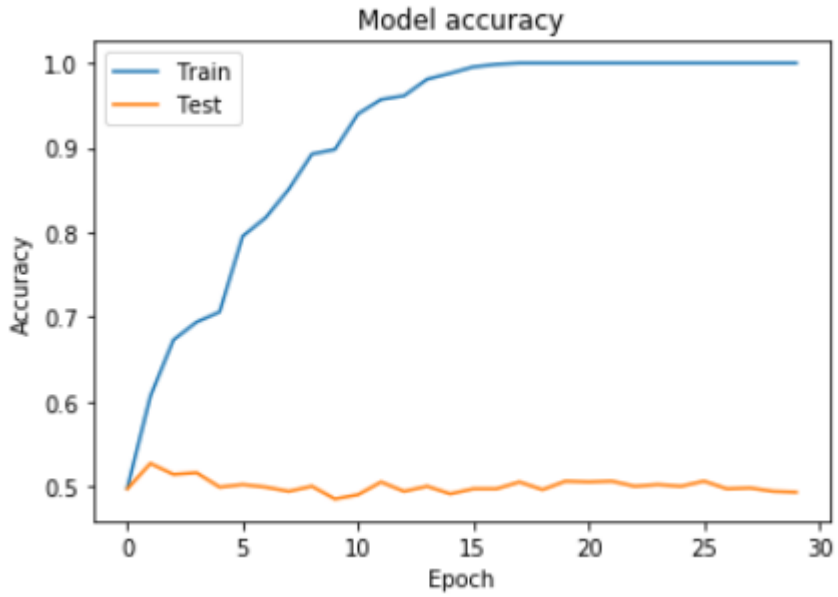
- Cambios para la siguiente prueba
  - Batch_size = 25
  - Capa oculta → 128 neuronas
  - Epochs = 60
  - Steps_per epoch = 1000
  - Validation_steps = 800

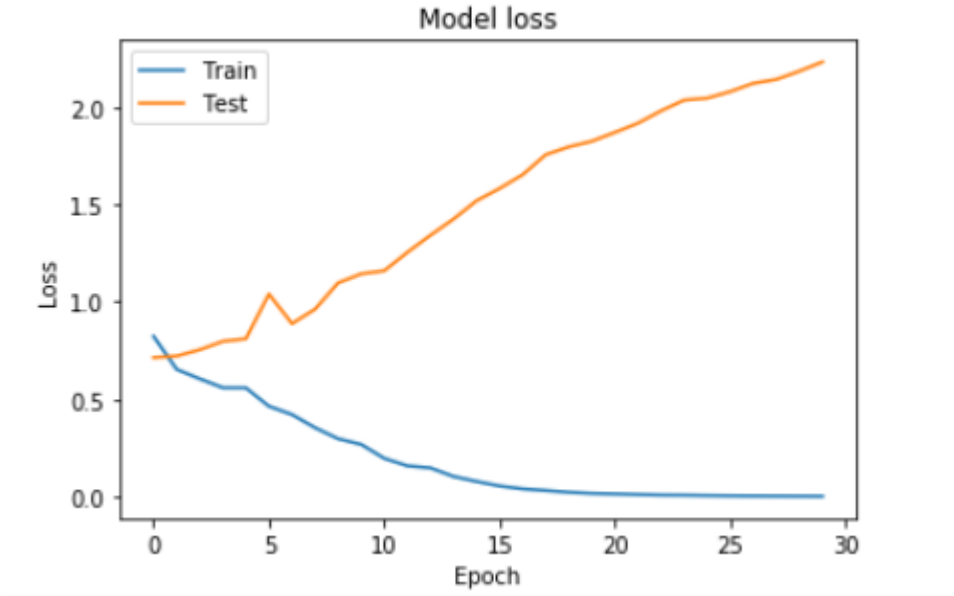




TRANSFER LEARNING

```
model = Sequential()
model.add(Flatten())
model.add(Dense(256, activation='relu'))
model.add(Dense(1, activation='sigmoid'))
```

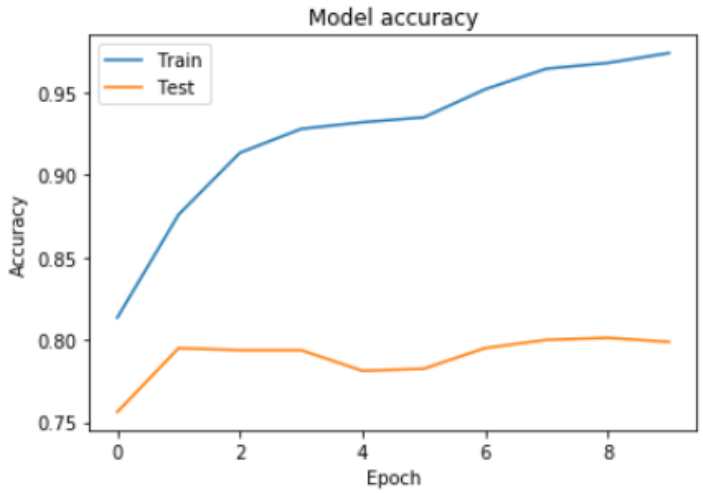


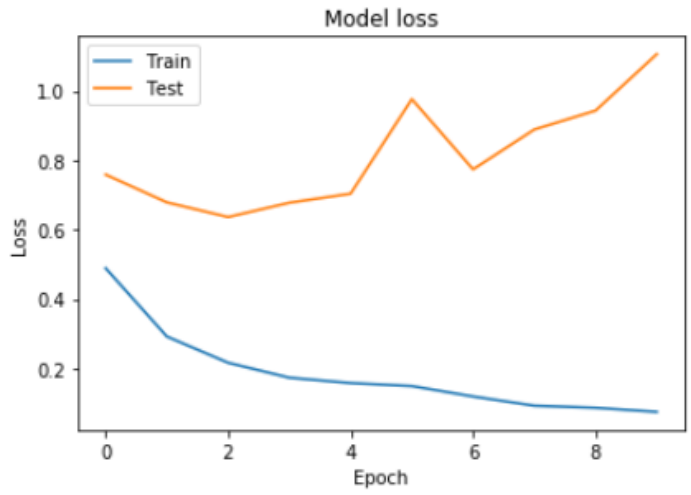


TRANSFER LEARNING

```
model = Sequential()
model.add(Flatten(input_shape=train_data.shape[1:]))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
nb_train_samples = 2000
nb_validation_samples = 800
epochs = 10
batch_size = 16
```

BASADO EN MODELO CAYETANO CON SU DATASET

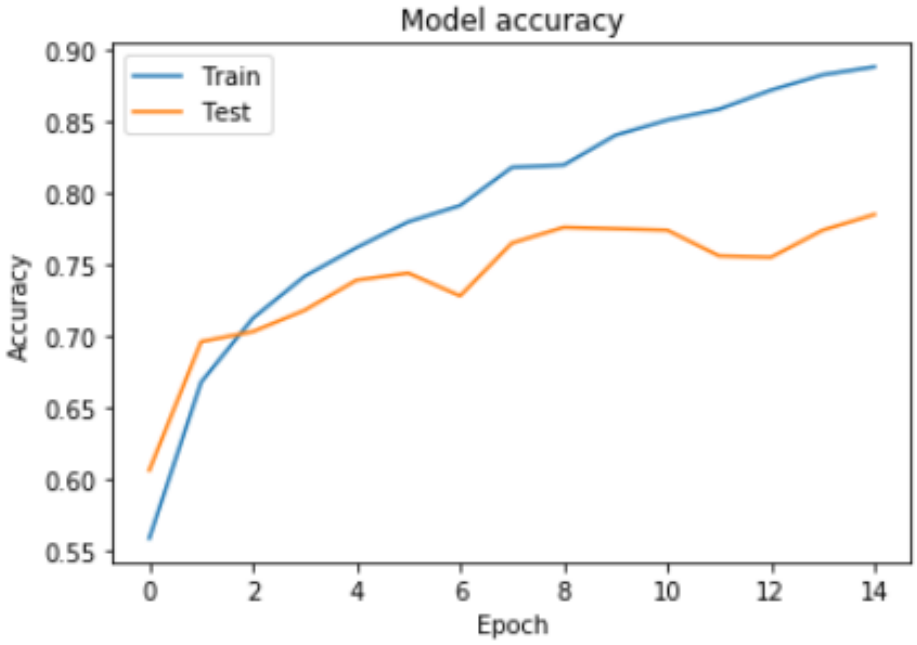


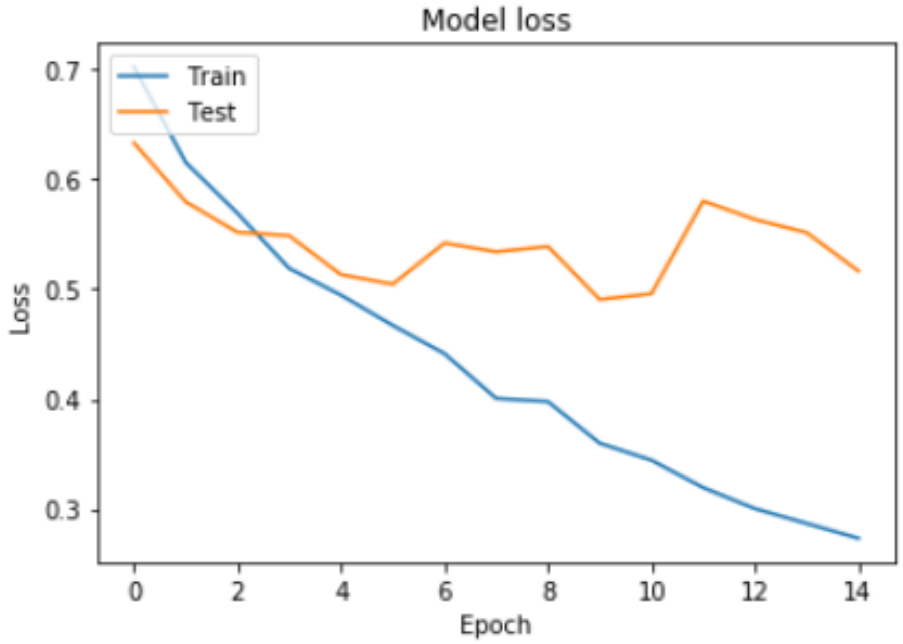


ÚLTIMA PRUEBA CON DATASET PERROS Y GATOS DE CAYETANO

- **Configuración sin Transfer Learning**

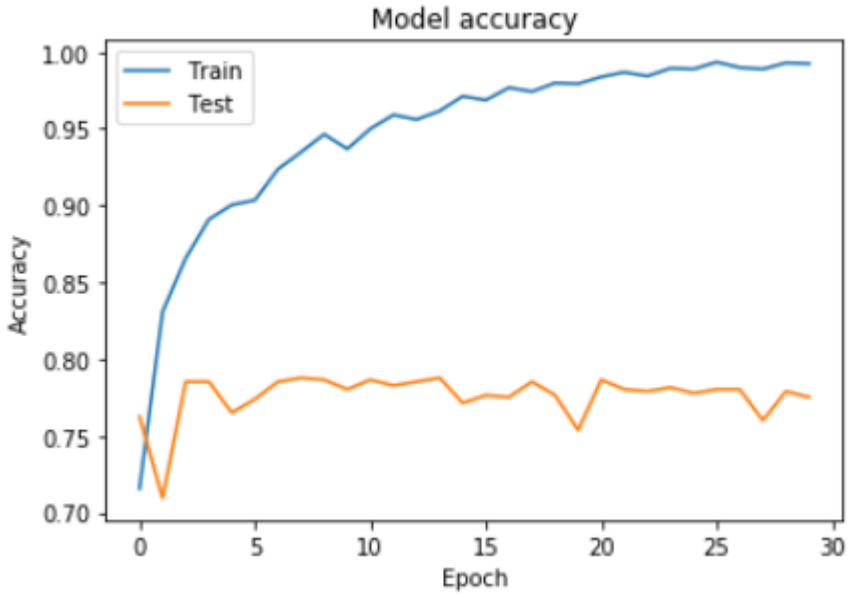
- Cambios para la siguiente prueba
  - Batch_size = 19
  - Capa oculta → 100 neuronas
  - Epochs = 15
  - Steps_per epoch = 300
  - Validation_steps = 800

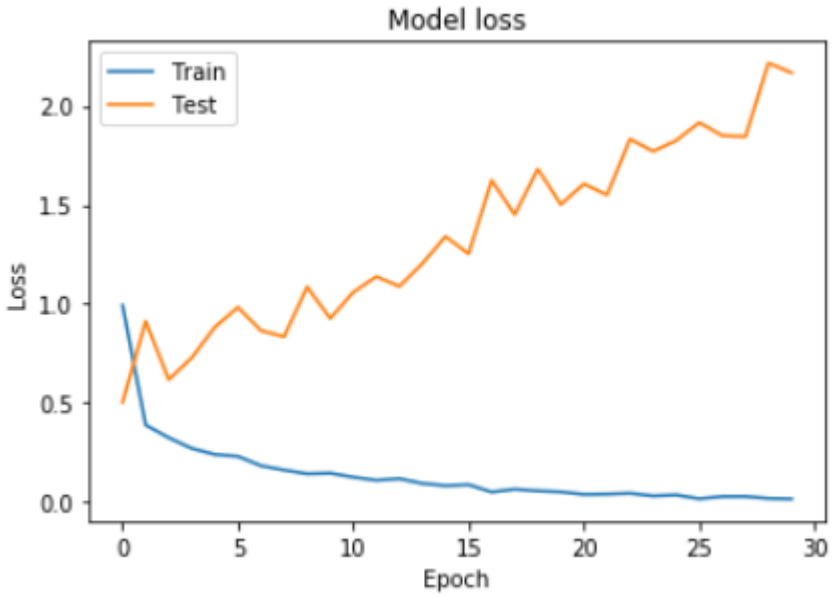


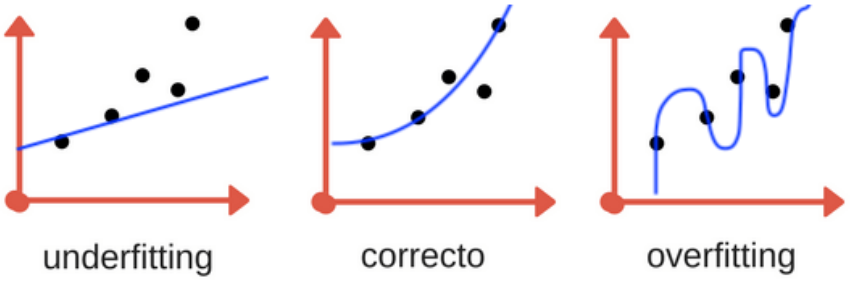


TRANSFER LEARNING – Casi Copia de la configuración de Cayetano

```
model = Sequential()
model.add(Flatten(input_shape=train_data.shape[1:]))
model.add(Dense(256, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))
nb_train_samples = 2000
nb_validation_samples = 800
epochs = 30
batch_size = 22
```







Las principales causas al obtener malos resultados en Machine Learning son el **overfitting o el underfitting de los datos**. Cuando entrenamos nuestro modelo intentamos «hacer encajar» -fit en inglés- los datos de entrada entre ellos y con la salida. Tal vez se pueda traducir overfitting como «sobreajuste» y underfitting como «subajuste» y *hacen referencia al fallo de nuestro modelo al generalizar* -encajar- el conocimiento que pretendemos que adquieran.

Prevenir el Sobreajuste de datos

Para intentar que estos problemas nos afecten lo menos posible, podemos llevar a cabo diversas acciones.

- **Cantidad mínima de muestras** tanto para entrenar el modelo como para validarlo.
- **Clases variadas y equilibradas en cantidad:** En caso de aprendizaje supervisado y suponiendo que tenemos que clasificar diversas clases o categorías, es importante que los datos de entrenamiento estén balanceados. Supongamos que tenemos que diferenciar entre manzanas, peras y bananas, debemos tener muchas fotos de las 3 frutas y en cantidades similares. Si tenemos muy pocas fotos de peras, esto afectará en el aprendizaje de nuestro algoritmo para identificar esa fruta.
- **Conjunto de validación de datos**. Siempre subdividir nuestro conjunto de datos y mantener una porción del mismo «oculto» a nuestra máquina entrenada. Esto nos permitirá obtener una valoración de aciertos/fallos real del modelo y también nos permitirá detectar fácilmente efectos del overfitting /underfitting.
- **Parameter Tunning o Ajuste de Parámetros:** deberemos experimentar sobre todo dando más/menos «tiempo/iteraciones» al entrenamiento y su aprendizaje hasta encontrar el equilibrio.
- **Cantidad excesiva de Dimensiones** (features), con muchas variantes distintas, sin suficientes muestras. A veces conviene eliminar o reducir la cantidad de características que utilizaremos para entrenar el modelo. Una herramienta útil para hacerlo es PCA.
- Podemos caer overfitting si usamos capas ocultas en exceso, ya que haríamos que el modelo memorice las posibles salidas, en vez de ser flexible y adecuar las activaciones a las entradas nuevas.

Si el modelo entrenado con el conjunto de test tiene un 90% de aciertos y con el conjunto de validación tiene un porcentaje muy bajo, esto señala claramente un problema de overfitting.

Si en el conjunto de validación sólo se acierta un tipo de clase (por ejemplo «peras») o el único resultado que se obtiene es siempre el mismo valor será que se produjo un problema de underfitting.