*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='sigmoid'),

      layers.Dense(64, activation='sigmoid'),

      layers.Dense(1)

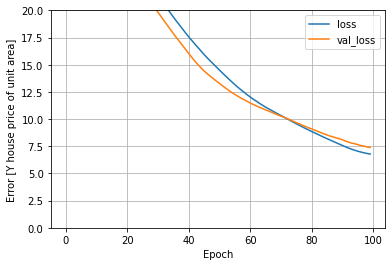
  ])

  model.compile(loss='mean\_absolute\_error',

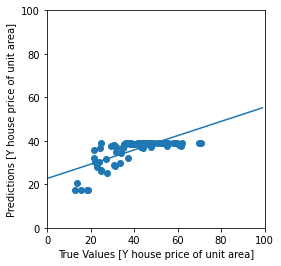
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

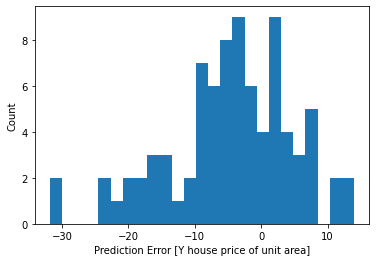
  return model

**RESULTS**

****

****

****

****

*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(1)

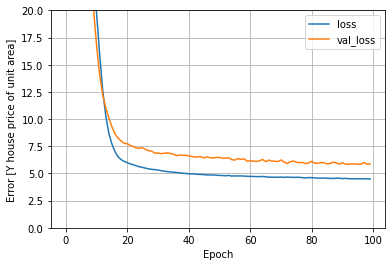
  ])

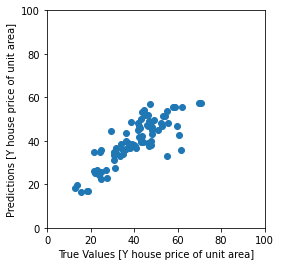
  model.compile(loss='mean\_absolute\_error',

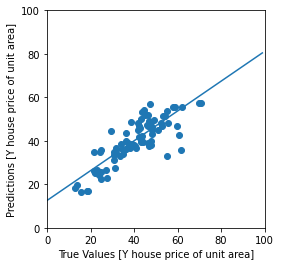
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

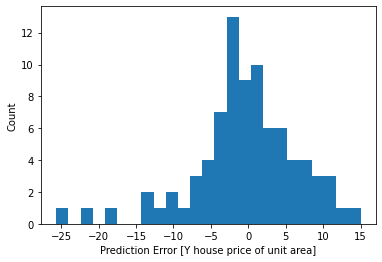
  return model

**RESULTS**









*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='tanh'),

      layers.Dense(64, activation='tanh'),

      layers.Dense(1)

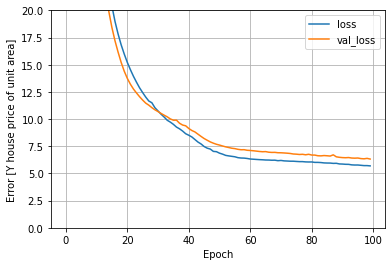
  ])

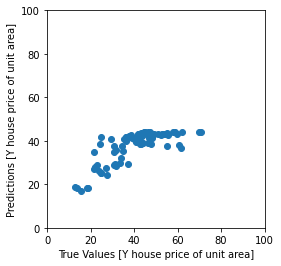
  model.compile(loss='mean\_absolute\_error',

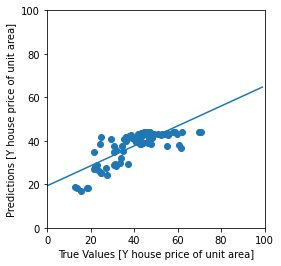
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

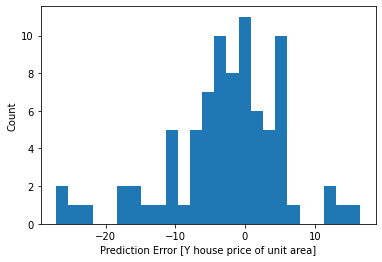
  return model

**RESULTS**









*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(1)

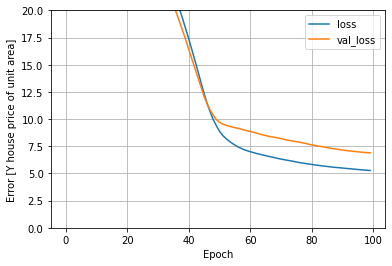
  ])

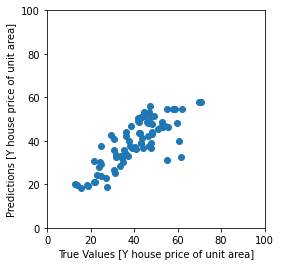
  model.compile(loss='mean\_absolute\_error',

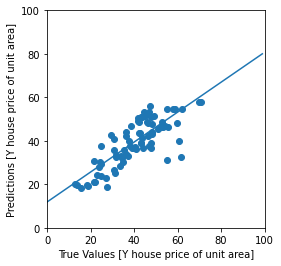
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

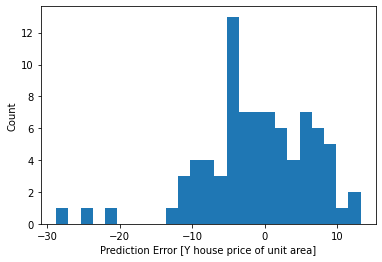
  return model

**RESULTS**









*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(1)

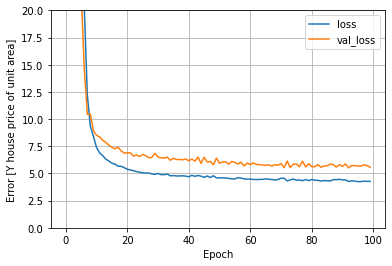
  ])

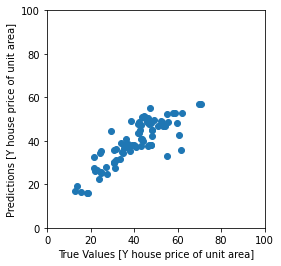
  model.compile(loss='mean\_absolute\_error',

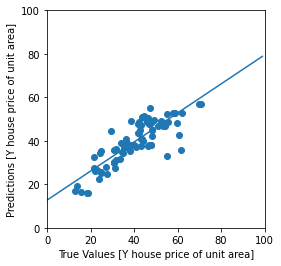
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

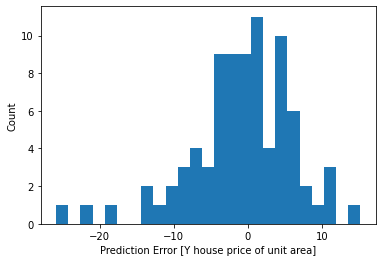
  return model

**RESULTS**

****

****

****

****

*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(1)

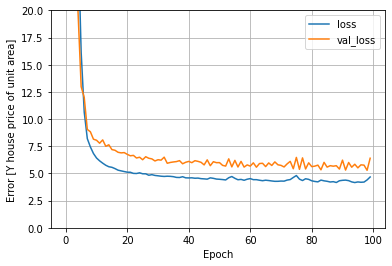
  ])

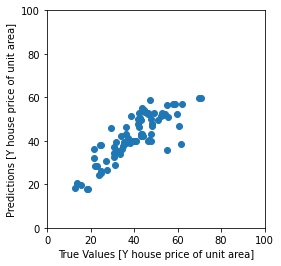
  model.compile(loss='mean\_absolute\_error',

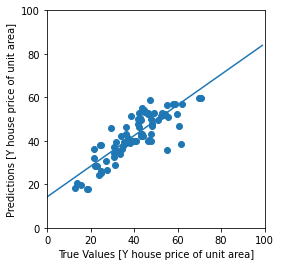
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

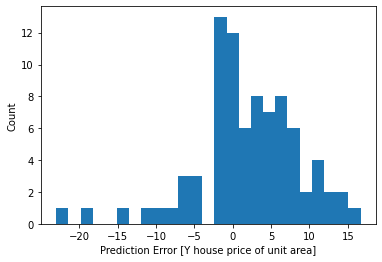
  return model

**RESULTS**

****

****

****

****

*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

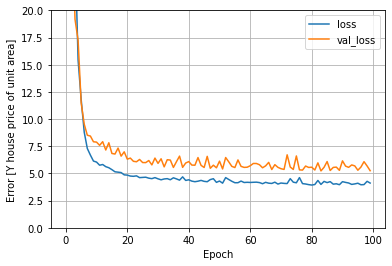
      layers.Dense(1)

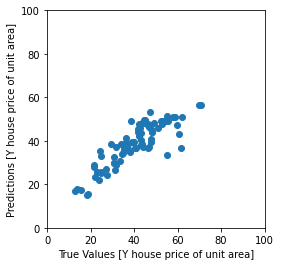
  ])

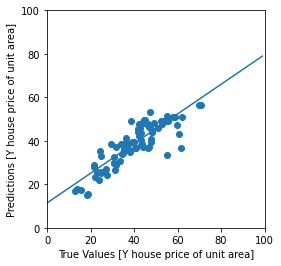
  model.compile(loss='mean\_absolute\_error',

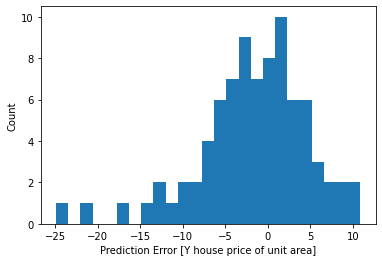
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

  return model

**RESULTS  
**

****

****

****

*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

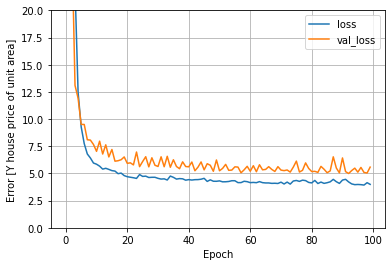
      layers.Dense(1)

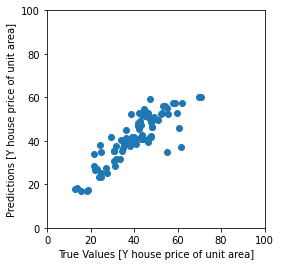
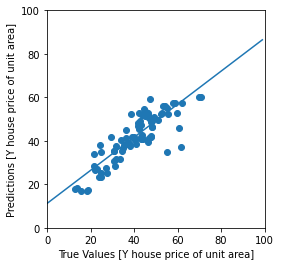
  ])

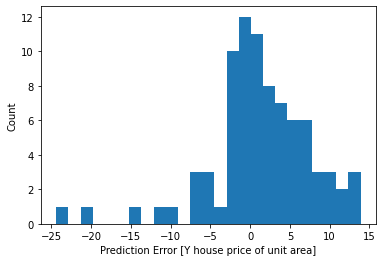
  model.compile(loss='mean\_absolute\_error',

                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

  return model

****

**  
**

****

*def* build\_and\_compile\_model(*norm*):

  model = keras.Sequential([

      norm,

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(64, activation='relu'),

      layers.Dense(1)

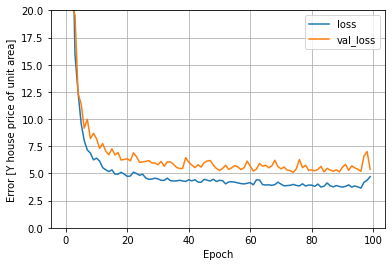
  ])

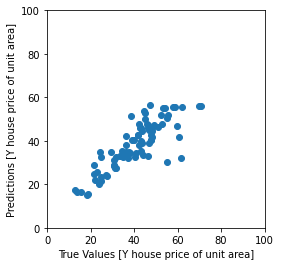
  model.compile(loss='mean\_absolute\_error',

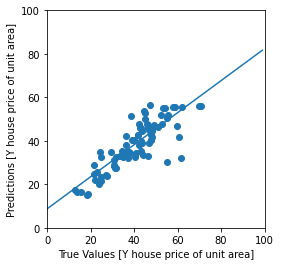
                optimizer=tf.keras.optimizers.Adam(0.001)) #loss = 'mse'

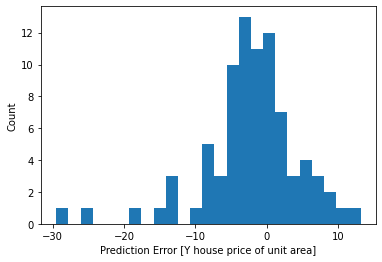
  return model

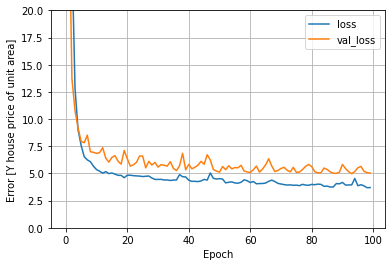
**RESULTS**

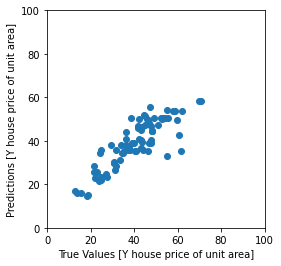
****

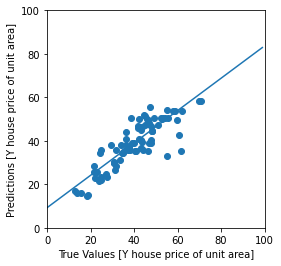
****

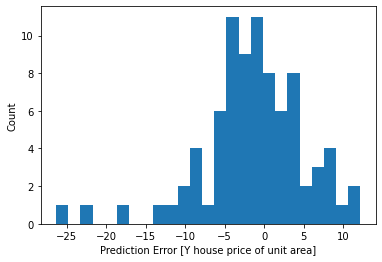
****

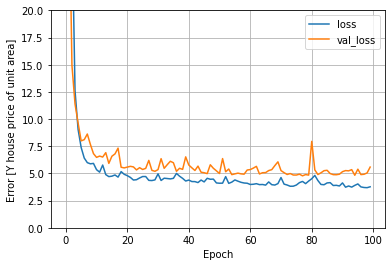
****

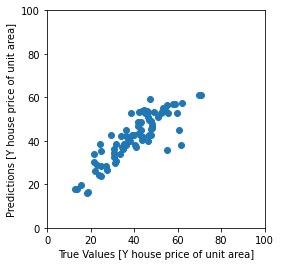
****

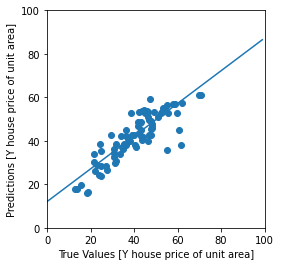
****

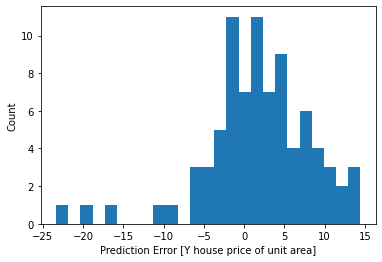
****

****

****

****

****

****