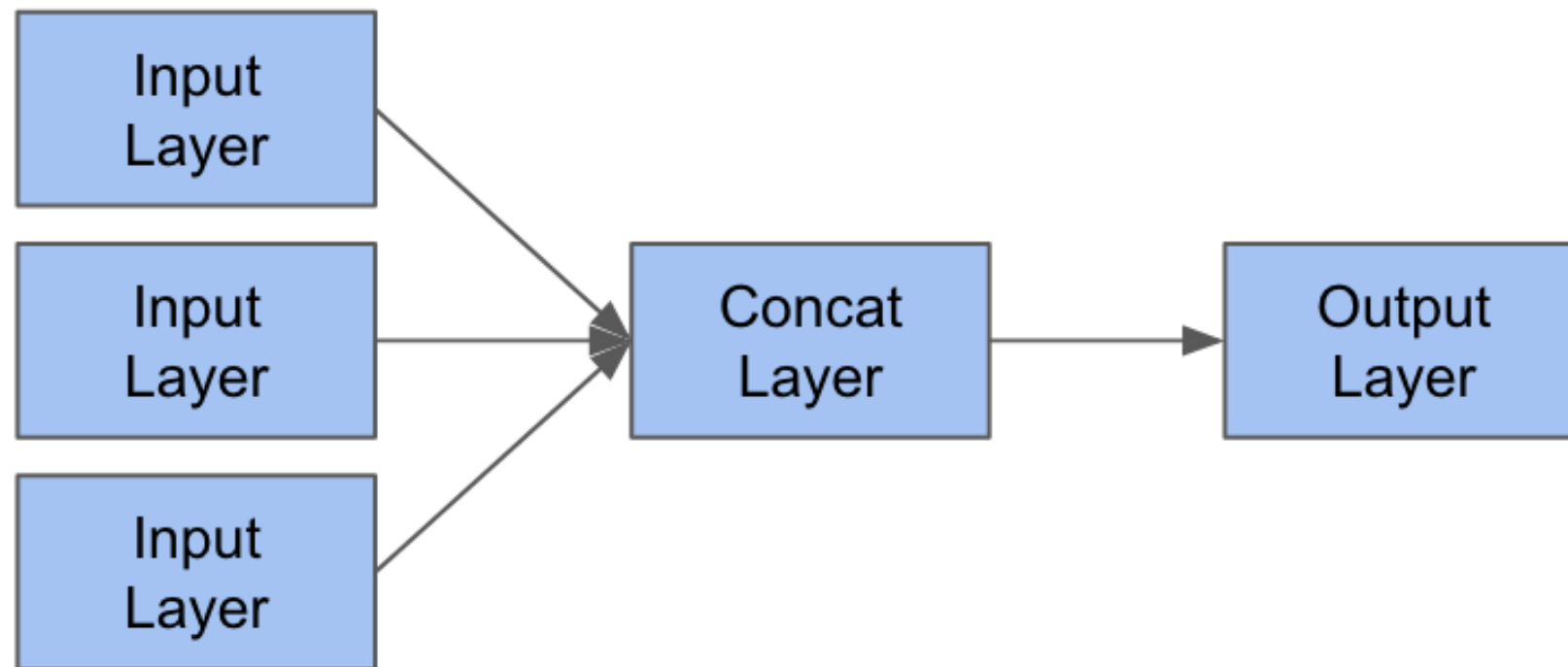# Three-input models

ADVANCED DEEP LEARNING WITH KERAS

**Zach Deane Mayer**
Data Scientist

Simple model with 3 inputs

```python
from keras.layers import Input, Concatenate, Dense
in_tensor_1 = Input(shape=(1,))
in_tensor_2 = Input(shape=(1,))
in_tensor_3 = Input(shape=(1,))
out_tensor = Concatenate()([in_tensor_1, in_tensor_2, in_tensor_3])
output_tensor = Dense(1)(out_tensor)
```

Simple model with 3 inputs

```
from keras.models import Model
model = Model([in_tensor_1, in_tensor_2, in_tensor_3], out_tensor)
```
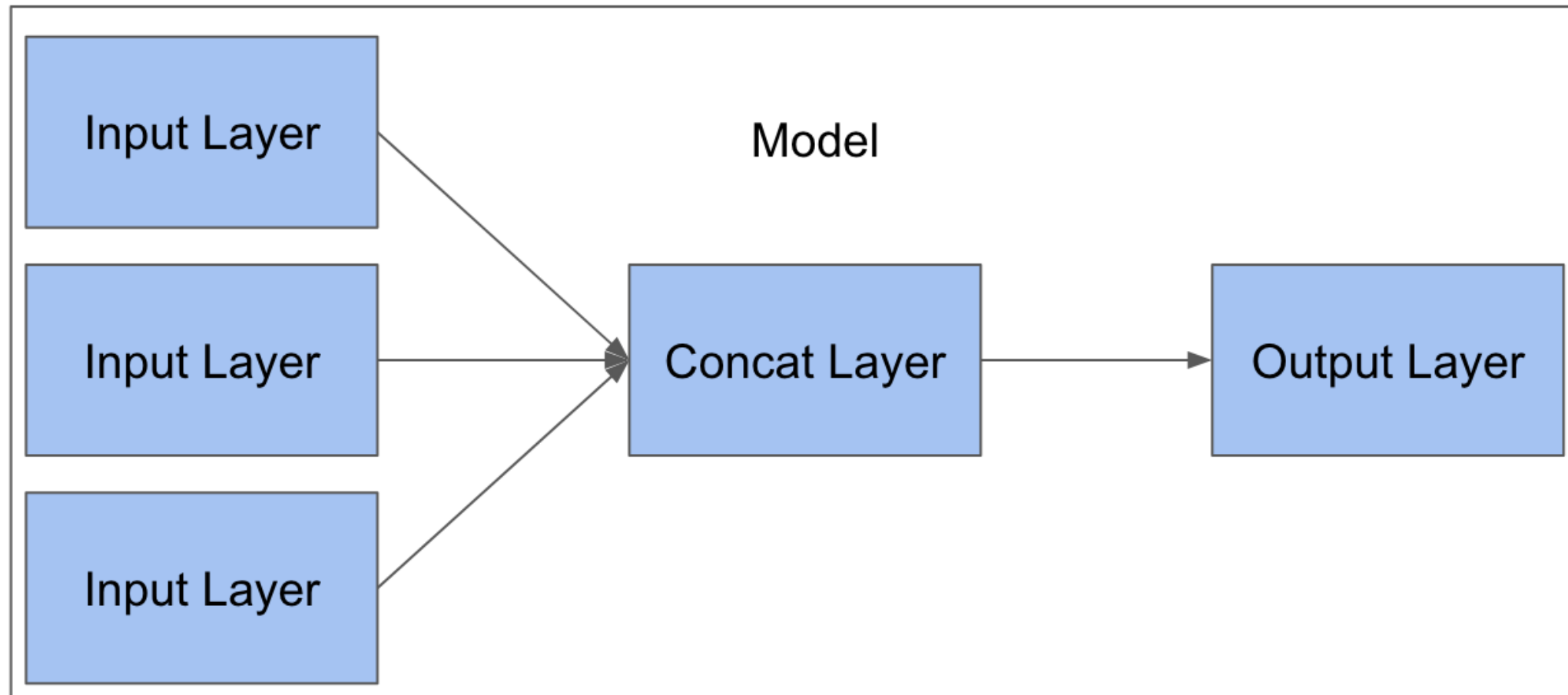
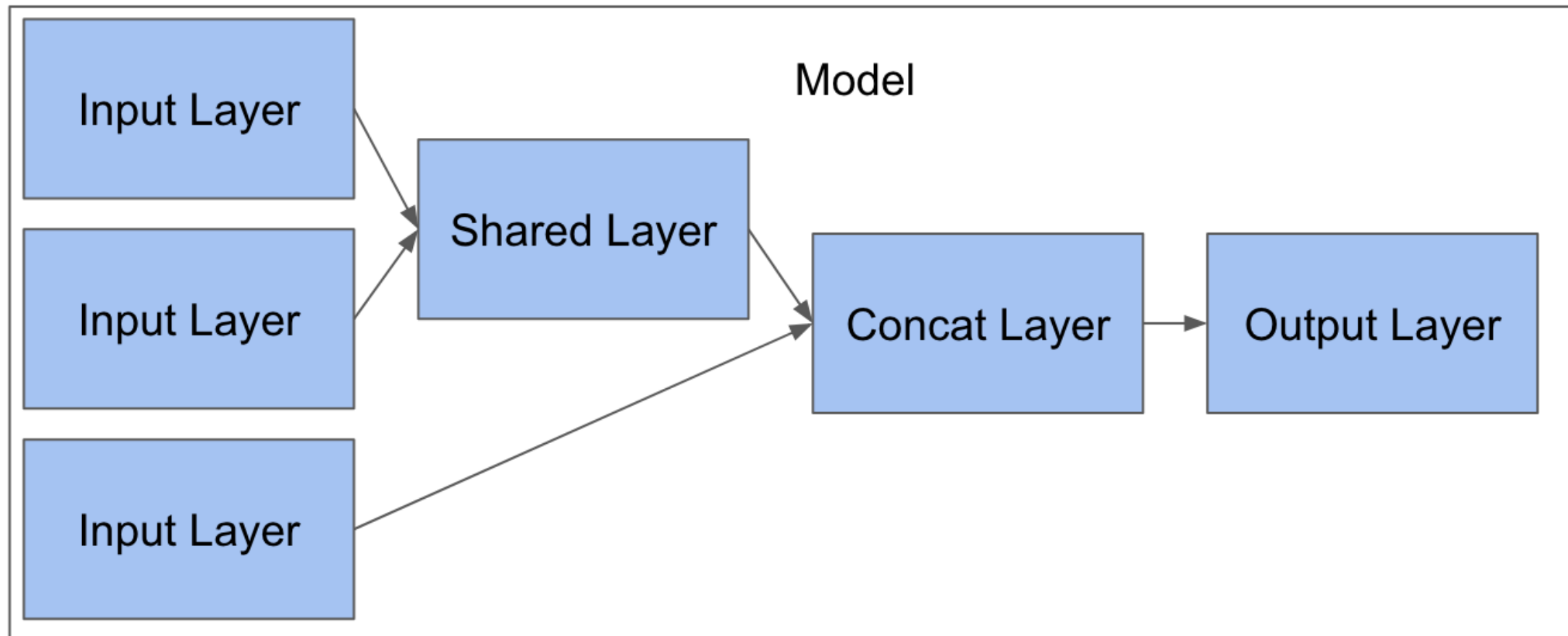# Shared layers with 3 inputs

```python
shared_layer = Dense(1)
shared_tensor_1 = shared_layer(in_tensor_1)
shared_tensor_2 = shared_layer(in_tensor_1)
out_tensor = Concatenate()([shared_tensor_1, shared_tensor_2, in_tensor_3])
out_tensor = Dense(1)(out_tensor)
```

Shared layers with 3 inputs

```
from keras.models import Model
model = Model([in_tensor_1, in_tensor_2, in_tensor_3], out_tensor)
```

Fitting a 3 input model

```
from keras.models import Model
model = Model([in_tensor_1, in_tensor_2, in_tensor_3], out_tensor)
model.compile(loss='mae', optimizer='adam')
```

```
model.fit([[train['col1'], train['col2'], train['col3']],
          train_data['target'])
```

```
model.evaluate([[test['col1'], test['col2'], test['col3']],
               test['target'])
```

Let's practice

ADVANCED DEEP LEARNING WITH KERAS

Summarizing and plotting models

ADVANCED DEEP LEARNING WITH KERAS

Zach Deane Mayer
Data Scientist

Understanding a model summary

```
--------------------------------------------------------------------------------
Layer (type)                    Output Shape          Param #      Connected to
================================================================================
input_1 (InputLayer)            (None, 1)             0

--------------------------------------------------------------------------------
input_2 (InputLayer)            (None, 1)             0

--------------------------------------------------------------------------------
input_3 (InputLayer)            (None, 1)             0

--------------------------------------------------------------------------------
concatenate_1 (Concatenate)     (None, 3)             0            input_1[0][0]
                                                                   input_2[0][0]
                                                                   input_3[0][0]

--------------------------------------------------------------------------------
dense_1 (Dense)                 (None, 1)             4            concatenate_1[0][0]
================================================================================
Total params: 4
Trainable params: 4
Non-trainable params: 0

--------------------------------------------------------------------------------
```
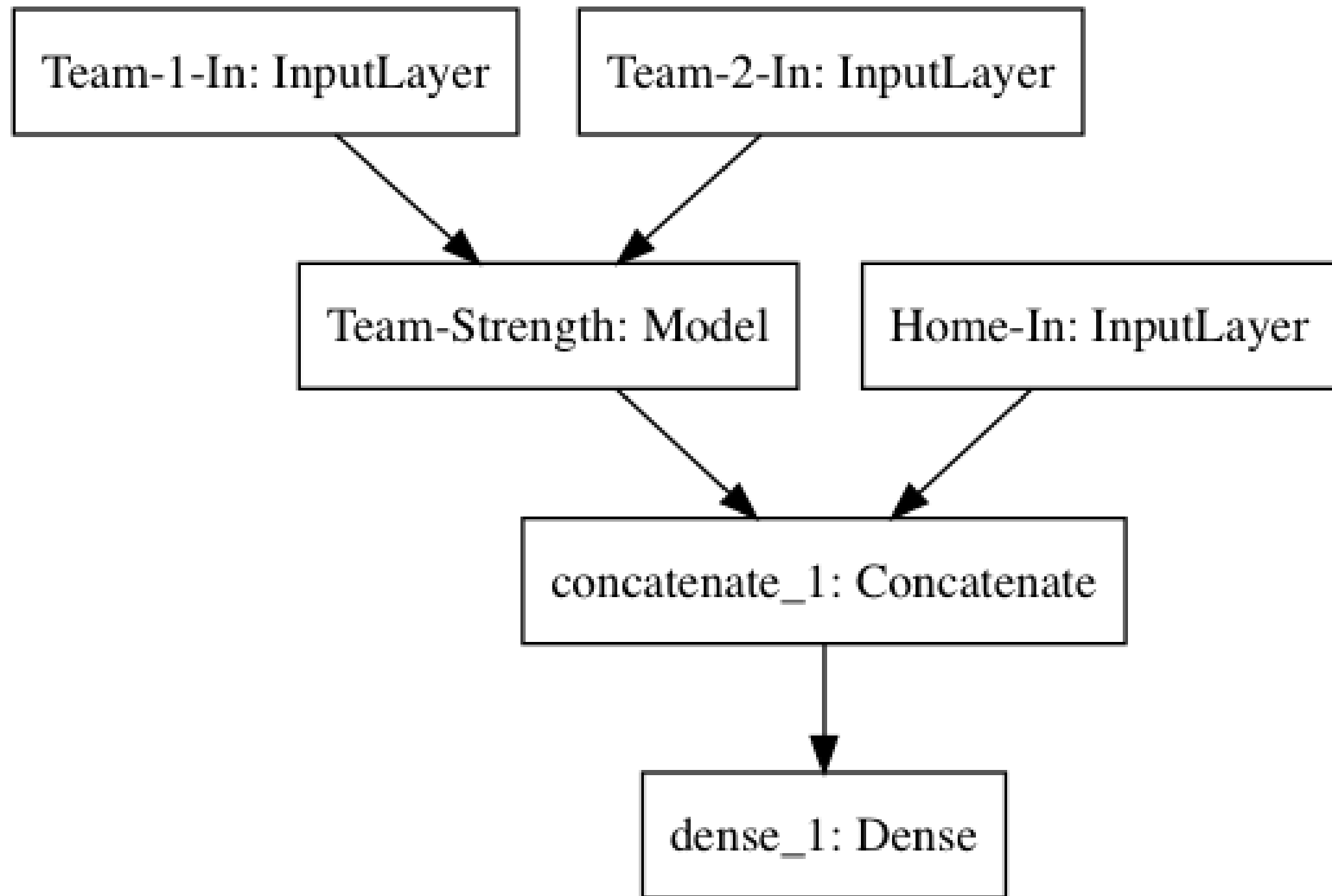
Understanding a model summary

```
_____
Layer (type)                    Output Shape         Param #     Connected to
==========================================================================
input_1 (InputLayer)            (None, 1)             0

_____
embedding_1 (Embedding)         (None, 1, 1)          10887       input_1[0][0]

_____
flatten_1 (Flatten)             (None, 1)             0           embedding_1[0][0]

_____
input_2 (InputLayer)            (None, 1)             0

_____
input_3 (InputLayer)            (None, 1)             0

_____
concatenate_1 (Concatenate)     (None, 3)             0           flatten_1[0][0]
                                                                  input_2[0][0]
                                                                  input_3[0][0]

_____
dense_1 (Dense)                 (None, 1)             4           concatenate_1[0][0]
==========================================================================
Total params: 10,891
Trainable params: 10,891
Non-trainable params: 0

_____
```
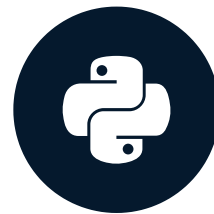
Understanding a model plot!
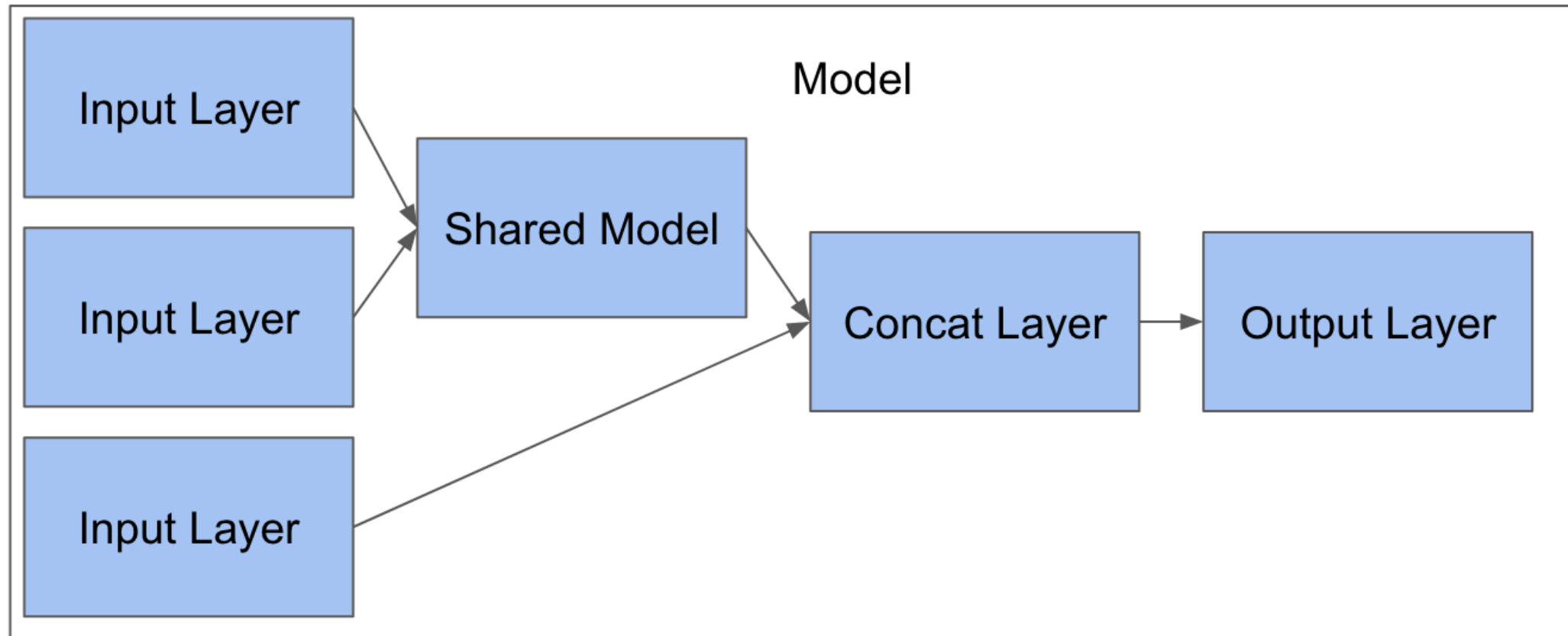
Let's Practice

ADVANCED DEEP LEARNING WITH KERAS

Stacking models

ADVANCED DEEP LEARNING WITH KERAS

Zach Deane Mayer
Data Scientist

Stacking models requires 2 datasets

```python
from pandas import read_csv
games_season = read_csv('datasets/games_season.csv')
games_season.head()
```

```
   team_1  team_2  home  score_diff
0    3745    6664     0          17
1     126    7493     1           7
2     288    3593     1           7
3    1846    9881     1          16
4    2675   10298     1          12
```

```python
games_tourney = read_csv('datasets/games_tourney.csv')
games_tourney.head()
```

```
   team_1  team_2  home  seed_diff  score_diff
0     288      73     0         -3          -9
1    5929      73     0          4           6
2    9884      73     0          5          -4
3      73     288     0          3           9
4    3920     410     0          1          -9
```

Enrich the tournament data

```python
in_data_1 = games_tourney['team_1']
in_data_2 = games_tourney['team_2']
in_data_3 = games_tourney['home']
pred = regular_season_model.predict([in_data_1, in_data_2, in_data_3])
```

```python
games_tourney['pred'] = pred
games_tourney.head()

   team_1  team_2  home  seed_diff       pred  score_diff
0     288      73     0         -3   0.582556          -9
1    5929      73     0          4   0.707279           6
2    9884      73     0          5   1.364844          -4
3      73     288     0          3   0.699145           9
4    3920     410     0          1   0.833066          -9
```
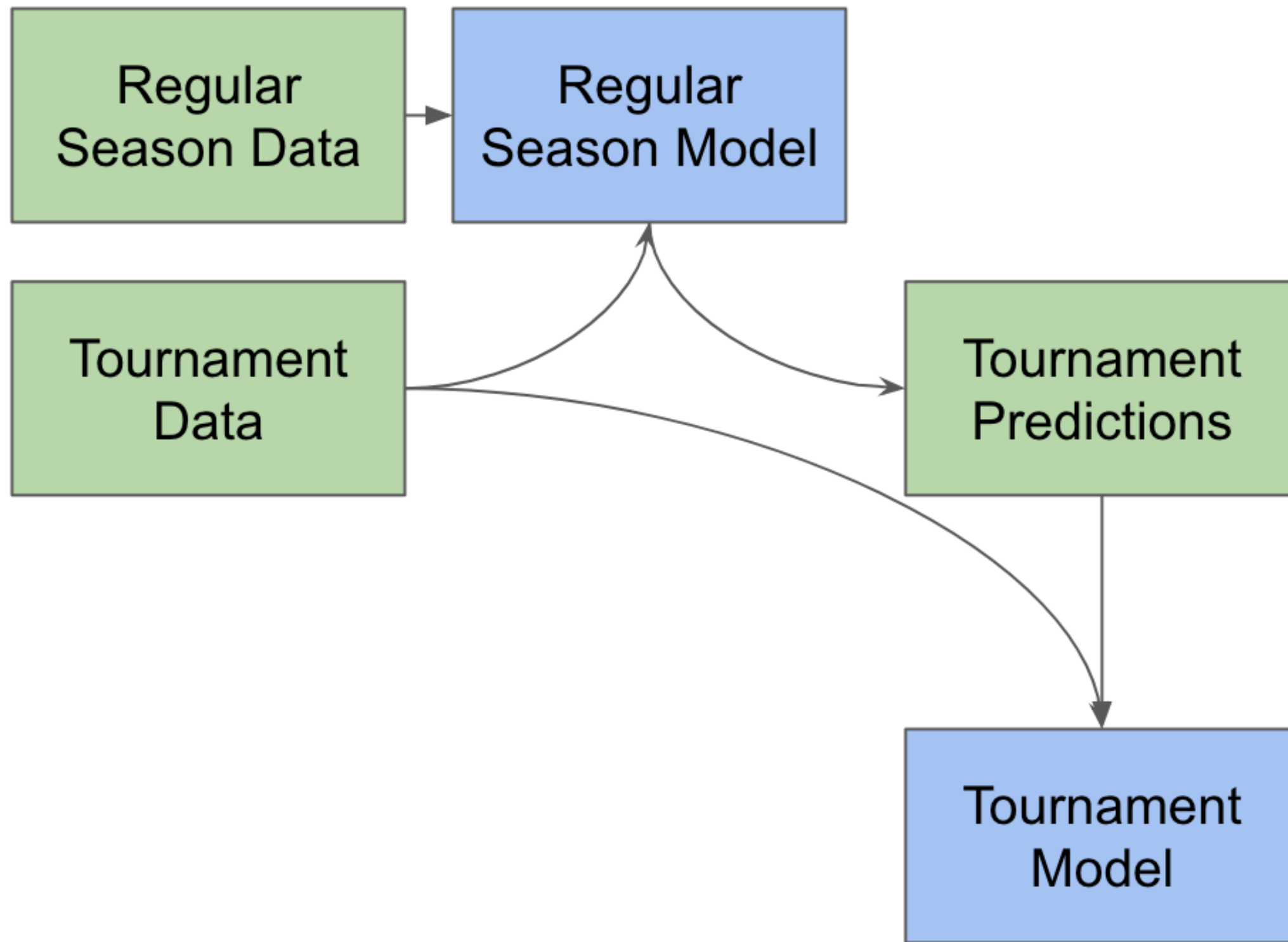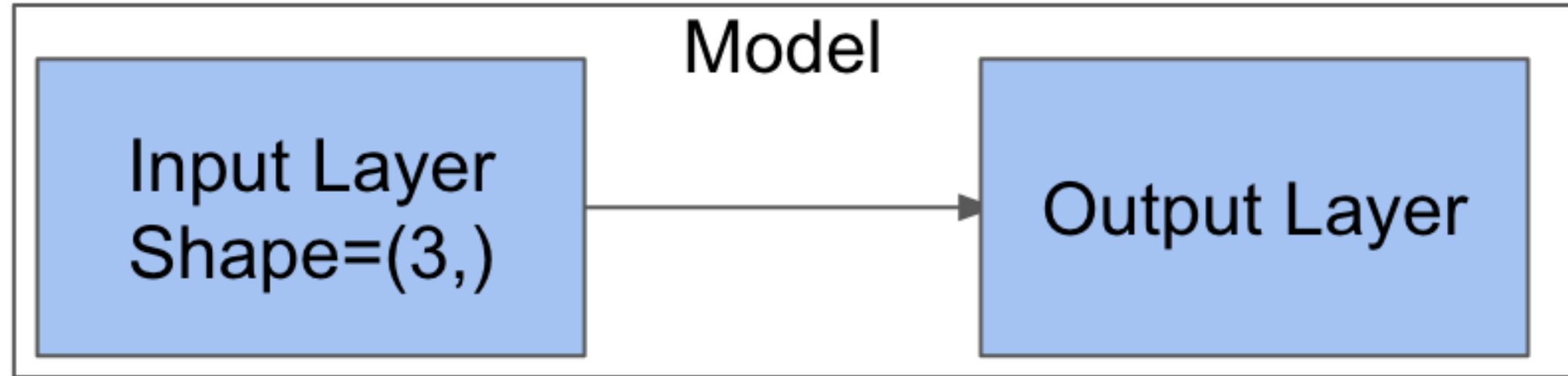
3 input model with pure numeric data

```
games_tourney[['home','seed_diff','pred']].head()


    home   seed_diff        pred
0      0          -3    0.582556
1      0           4    0.707279
2      0           5    1.364844
3      0           3    0.699145
4      0           1    0.833066
```

3 input model with pure numeric data

3 input model with pure numeric data

```python
from keras.layers import Input, Dense
in_tensor = Input(shape=(3,))
out_tensor = Dense(1)(in_tensor)
```

```python
from keras.models import Model
model = Model(in_tensor, out_tensor)
model.compile(optimizer='adam', loss='mae')
train_X = train_data[['home','seed_diff','pred']]
train_y = train_data['score_diff']
model.fit(train_X,train_y, epochs=10, validation_split=.10)
```

```python
test_X = test_data[['home','seed_diff','pred']]
test_y = test_data['score_diff']
model.evaluate(test_X, test_y)
1066/1066 [==============================] - 0s 14us/step
9.11321775461451
```

Let's practice!

ADVANCED DEEP LEARNING WITH KERAS