

# Keras input and dense layers

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON



**Zach Deane-Mayer**  
Data Scientist

# Course outline

- Chapter 1: Introduction to the Keras functional API (Refresher)
- Chapter 2: Models with 2 inputs
- Chapter 3: Models with 3 inputs
- Chapter 4: Multiple outputs

# Course Datasets: College basketball data, 1989-2017

## Dataset 1: Regular season

- Team ID 1
- Team ID 2
- Home vs Away
- Score Difference (Team 1 - Team 2)
- Team 1 Score
- Team 2 Score

## Dataset 2: Tournament games

- Same as Dataset 1
- Also has difference in Seed

# Course Datasets: College basketball data, 1989-2017

```
import pandas as pd
games_season = pd.read_csv('datasets/games_season.csv')
games_season.head()
```

Out[1]:

	season	team_1	team_2	home	score_diff	score_1	score_2	won
0	1985	3745	6664	0	17	81	64	1
1	1985	126	7493	1	7	77	70	1
2	1985	288	3593	1	7	63	56	1
3	1985	1846	9881	1	16	70	54	1
4	1985	2675	10298	1	12	86	74	1

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

Out[2]:

	season	team_1	team_2	home	seed_diff	score_diff	score_1	score_2	won
0	1985	288	73	0	-3	-9	41	50	0
1	1985	5929	73	0	4	6	61	55	1
2	1985	9884	73	0	5	-4	59	63	0
3	1985	73	288	0	3	9	50	41	1
4	1985	3920	410	0	1	-9	54	63	0

# Inputs and outputs

Two fundamental parts:

- Input layer
- Output layer



# Inputs

```
from keras.layers import Input  
input_tensor = Input(shape=(1,))
```



# Inputs

```
from keras.layers import Input
input_tensor = Input(shape=(1,))
print(input_tensor)

<tf.Tensor 'input_1:0' shape=(?, 1) dtype=float32>
```



# Outputs

```
from keras.layers import Dense  
output_layer = Dense(1)
```

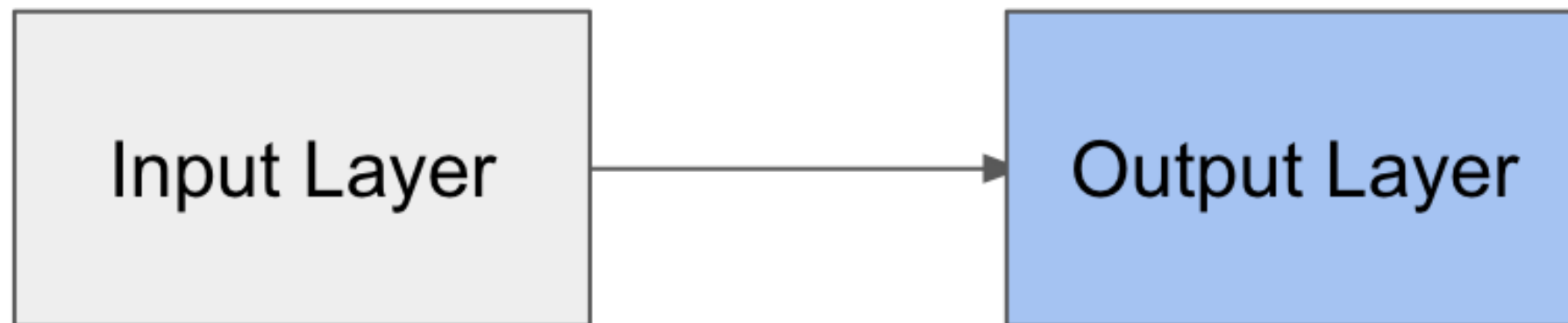




# Outputs

```
from keras.layers import Dense
output_layer = Dense(1)
print(output_layer)

<keras.layers.core.Dense at 0x7f22e0295a58>
```



# Connecting inputs to outputs

```
from keras.layers import Input, Dense
input_tensor = Input(shape=(1,))
output_layer = Dense(1)
output_tensor = output_layer(input_tensor)
```



# Connecting inputs to outputs

```
print(output_tensor)
```

```
<tf.Tensor 'dense_1/BiasAdd:0' shape=(?, 1) dtype=float32>
```



# Let's practice!

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON

# Keras models

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON



**Zach Deane-Mayer**  
Data Scientist

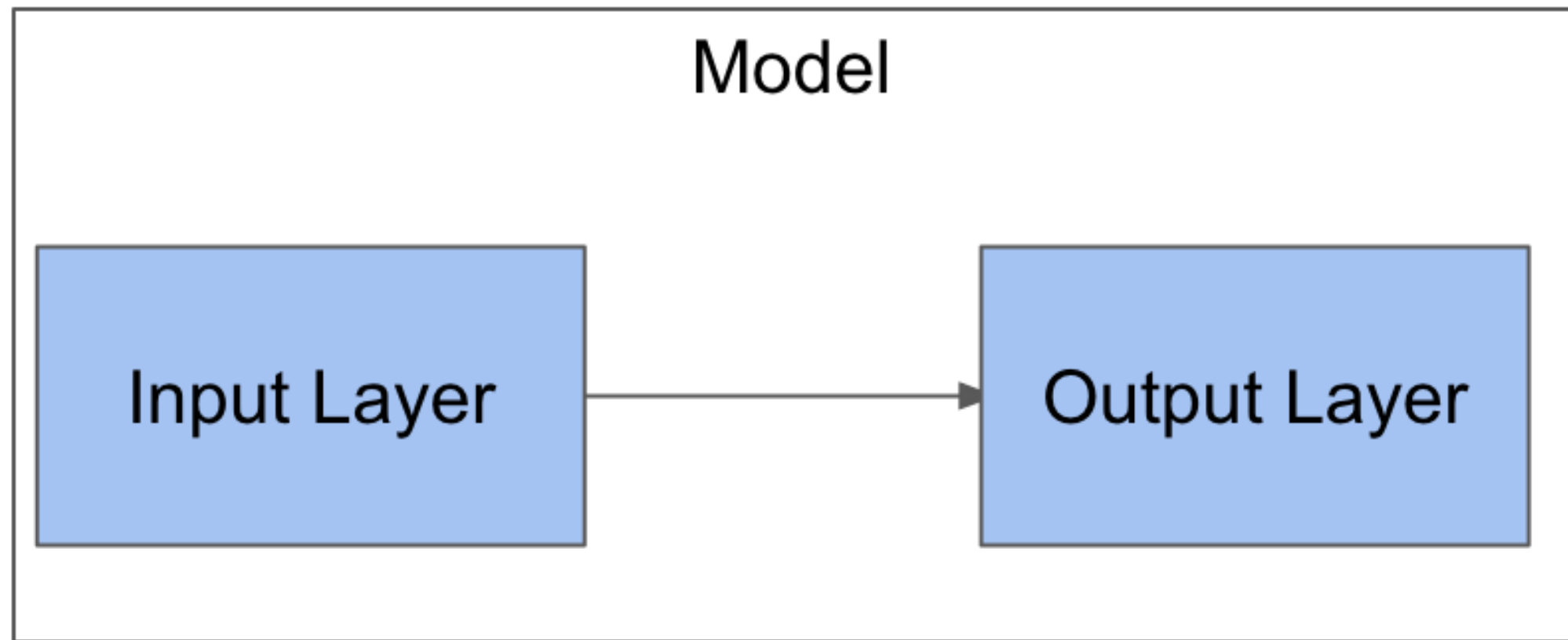
# Keras models

```
from keras.layers import Input, Dense  
input_tensor = Input(shape=(1,))  
output_tensor = Dense(1)(input_tensor)
```



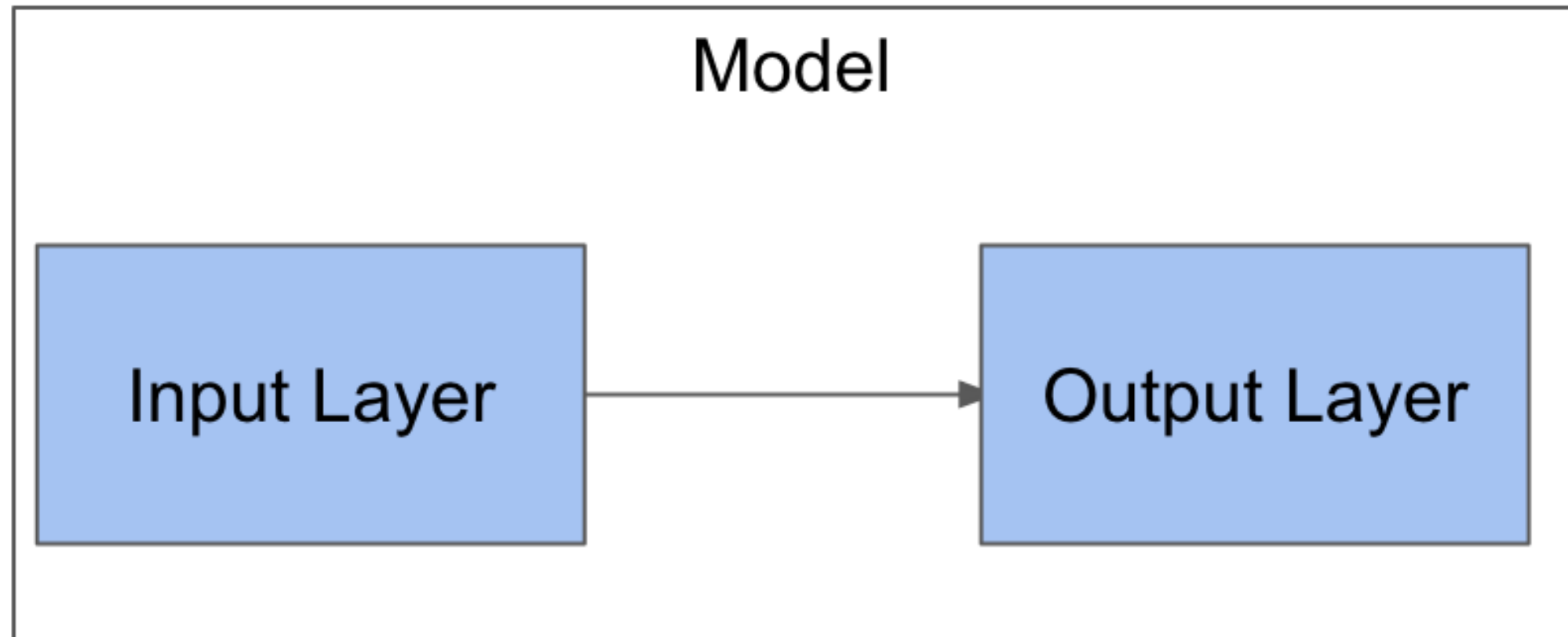
# Keras models

```
from keras.models import Model  
model = Model(input_tensor, output_tensor)
```



# Compile a model

```
model.compile(optimizer='adam', loss='mae')
```





# Summarize the model

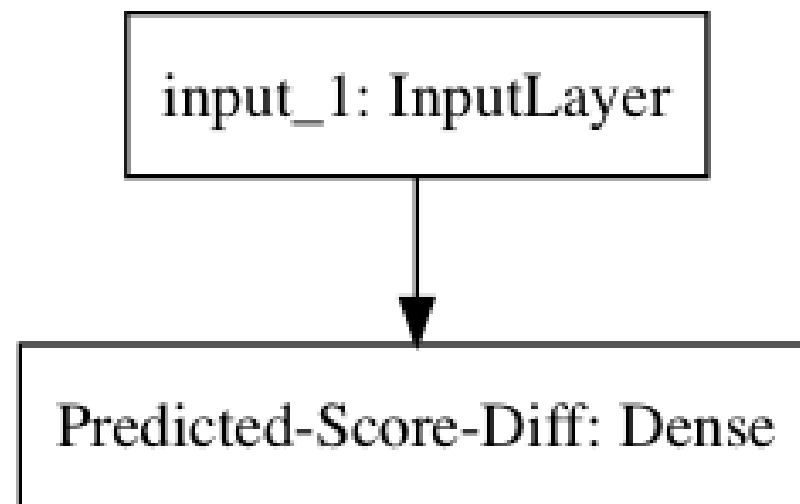
```
model.summary()
```

```
-----  
Layer (type)                 Output Shape                 Param #  
-----  
input_1 (InputLayer)         (None, 1)                   0  
-----  
dense_1 (Dense)              (None, 1)                   2  
-----  
Total params: 2  
Trainable params: 2  
Non-trainable params: 0  
-----
```

# Plot model using keras

```
input_tensor = Input(shape=(1,))
output_layer = Dense(1, name='Predicted-Score-Diff')
output_tensor = output_layer(input_tensor)
model = Model(input_tensor, output_tensor)
plot_model(model, to_file='model.png')

from matplotlib import pyplot as plt
img = plt.imread('model.png')
plt.imshow(img)
plt.show()
```



# Let's practice!

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON

# Fit and evaluate a model

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON



Zach Deane-Mayer  
Data Scientist

# Basketball Data

Goal: Predict tournament outcomes

Data Available: team ratings from the tournament organizers

```
import pandas as pd
games_tourney = pd.read_csv('datasets/games_tourney.csv')
games_tourney.head()
```

Out[1]:

	season	team_1	team_2	home	seed_diff	score_diff	score_1	score_2	won
0	1985	288	73	0	-3	-9	41	50	0
1	1985	5929	73	0	4	6	61	55	1
2	1985	9884	73	0	5	-4	59	63	0
3	1985	73	288	0	3	9	50	41	1
4	1985	3920	410	0	1	-9	54	63	0

# Basketball Data

Input: Seed difference

```
import pandas as pd
games_tourney = pd.read_csv('datasets/games_tourney.csv')
games_tourney.head()
```

Out[1]:

	season	team_1	team_2	home	seed_diff	score_diff	score_1	score_2	won
0	1985	288	73	0	-3	-9	41	50	0
1	1985	5929	73	0	4	6	61	55	1
2	1985	9884	73	0	5	-4	59	63	0
3	1985	73	288	0	3	9	50	41	1
4	1985	3920	410	0	1	-9	54	63	0

# Basketball Data

Output: Score difference

```
import pandas as pd
games_tourney = pd.read_csv('datasets/games_tourney.csv')
games_tourney.head()
```

Out[1]:

	season	team_1	team_2	home	seed_diff	score_diff	score_1	score_2	won
0	1985	288	73	0	-3	-9	41	50	0
1	1985	5929	73	0	4	6	61	55	1
2	1985	9884	73	0	5	-4	59	63	0
3	1985	73	288	0	3	9	50	41	1
4	1985	3920	410	0	1	-9	54	63	0

# Basketball Data

Input:

- Seed difference - one number: -15 to +15
- Seed range from 1-16
- Highest difference is  $16-1 = +15$
- Lowest difference is  $1-16 = -15$

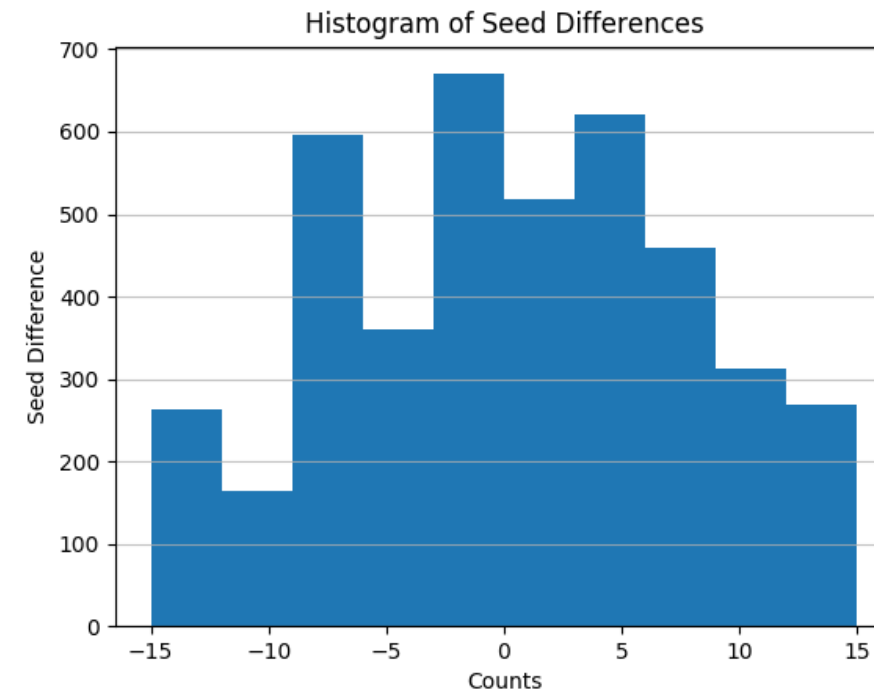
Output:

- Score difference - one number: -50 to +50



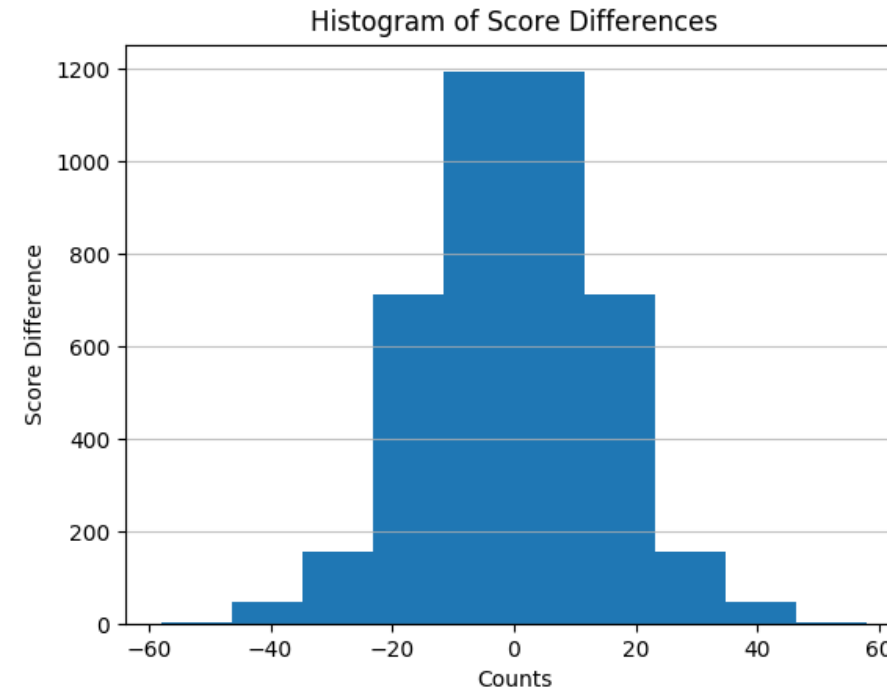
# Basketball Data

- Seed difference: 15
  - Team 1: 16
  - Team 2: 1
- Seed difference: -15
  - Team 1: 1
  - Team 2: 16



# Basketball Data

- Score difference: -9
  - Team 1: 41
  - Team 2: 50
- Score difference: 6
  - Team 1: 61
  - Team 2: 55



# Basketball Data

```
import pandas as pd
games_tourney = pd.read_csv('datasets/games_tourney_samp.csv')
games_tourney.head()
```

Out[1]:

	season	team_1	team_2	home	seed_diff	score_diff	score_1	score_2	won
0	2017	320	6323	0	13	18	100	82	1
1	2017	6323	320	0	-13	-18	82	100	0

# Build the model

```
from keras.models import Model
from keras.layers import Input, Dense
input_tensor = Input(shape=(1,))
output_tensor = Dense(1)(input_tensor)
model = Model(input_tensor, output_tensor)
model.compile(optimizer='adam', loss='mae')
```



# Fit the model

```
from pandas import read_csv
games = read_csv('datasets/games_tourney.csv')
model.fit(games['seed_diff'],
          games['score_diff'],
          batch_size=64,
          validation_split=.20,
          verbose=True)
```



# Evaluate the model

```
model.evaluate(games_test['seed_diff'],
               games_test['score_diff'])

1000/1000 [=====] - 0s 26us/step
Out[1]: 9.742335235595704
```

# Let's practice!

ADVANCED DEEP LEARNING WITH KERAS IN PYTHON