

# Deep Learning in R



Обзор фреймворков с примерами

metya

2018-12-08

# Disclaimer

Цель доклада не дать понимание что такое глубокое обучение и детально разобрать как работать с ним и обучать современные модели, а скорее показать как просто можно начать тем, кто давно хотел и чесались руки, но все было никак не взяться

# Deep Learning

Что это?

# Deep Learning

## Что это?

- Когда у нас есть искусственная нейронная сеть

# Deep Learning

## Что это?

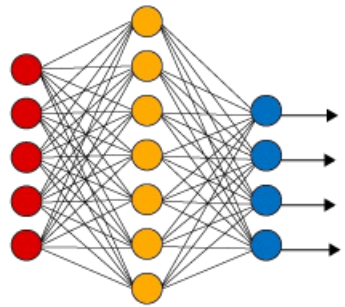
- Когда у нас есть искусственная нейронная сеть
- Когда скрытых слоев в этой сети больше чем два

# Deep Learning

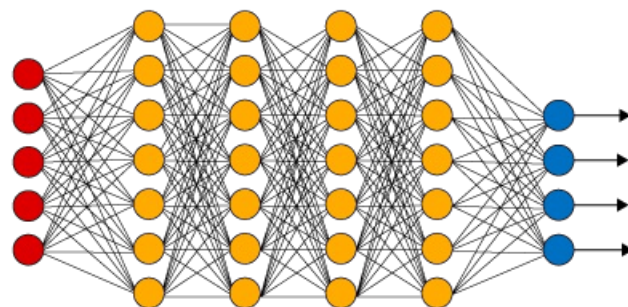
## Что это?

- Когда у нас есть искусственная нейронная сеть
- Когда скрытых слоев в этой сети больше чем два

**Simple Neural Network**



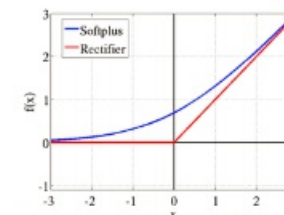
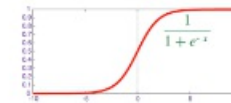
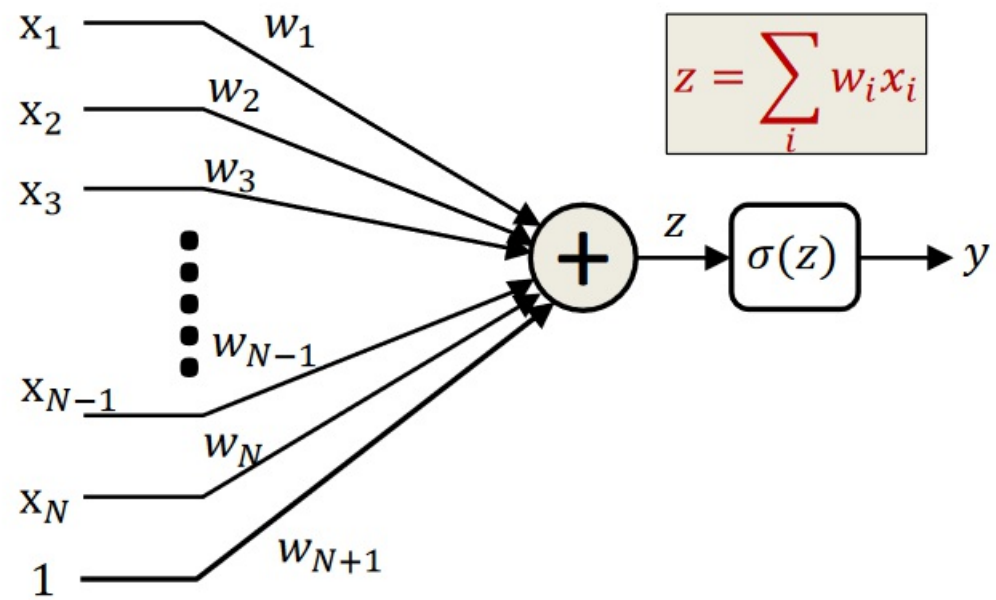
**Deep Learning Neural Network**



● Input Layer    ● Hidden Layer    ● Output Layer

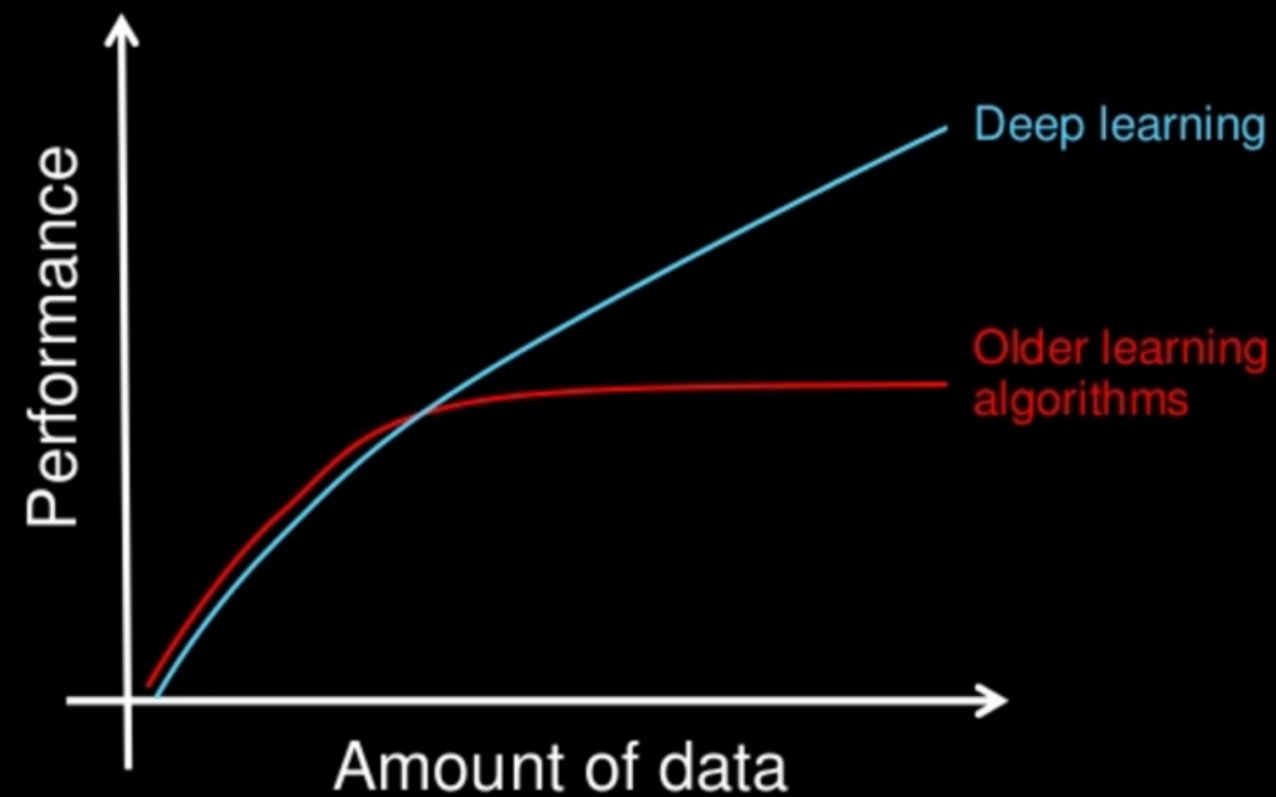
[1] <https://machinelearningmastery.com/what-is-deep-learning/>

## Как это математически



Activation functions  $\sigma(z)$

# Why deep learning

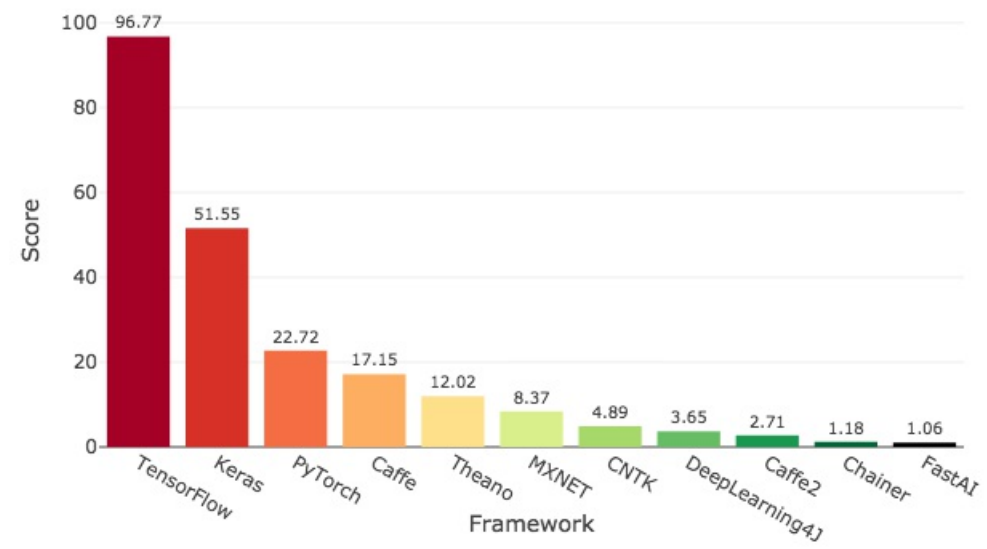


How do data science techniques scale with amount of data?



# Frameworks

Deep Learning Framework Power Scores 2018



[1] <https://towardsdatascience.com/deep-learning-framework-power-scores-2018-23607ddf297a>

Нас интересуют только те, что есть в R через API

# Нас интересуют только те, что есть в R через API

- TensorFlow

# Нас интересуют только те, что есть в R через API

- TensorFlow
- theano

## Нас интересуют только те, что есть в R через API

- TensorFlow
- theano
- Keras

## Нас интересуют только те, что есть в R через API

- TensorFlow
- theano
- Keras
- CNTK

## Нас интересуют только те, что есть в R через API

- TensorFlow
- theano
- Keras
- CNTK
- MXNet



## Нас интересуют только те, что есть в R через API

- TensorFlow
- theano
- Keras
- CNTK
- MXNet
- ONNX

## Есть еще несколько пакетов

- darch (removed from cran)
- deepnet
- deepr
- H2O (interface) ([Tutorial](#))



<https://www.tensorflow.org/>

<https://tensorflow.rstudio.com/>

- Делает Google
- Самый популярный, имеет тучу туториалов и книг
- Имеет самый большой спрос у продакшн систем
- Имеет API во множестве языков
- Имеет статический граф вычислений, что бывает неудобно, зато оптимизированно
- Примерно с лета имеет фичу **eager execution**, который почти нивелирует это неудобство. Но почти не считается
- Доступен в R как самостоятельно, так и как бэкэнд Keras

# theano

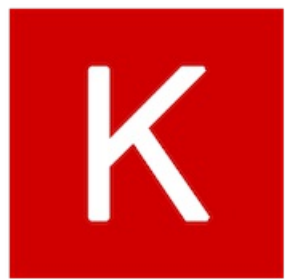
<http://www.deeplearning.net/software/theano/>

- Делался силами университета Монреаль с 2007
- Один из самых старых фреймворков, но почти похоронен в забвении
- Придумали идею абстракции вычислительных графов (статических) для оптимизации и вычисления нейронных сетей
- В R доступен как бэкенд через Keras



<https://cntk.ai/>

- Делается силами Майкрософт
- Имеет половинчатые динамические вычислительные графы (на самом деле динамические тензоры скорее)
- Доступен как бэкенд Keras так и как самостоятельный бэкенд с биндингами в R через reticulate package, что значит нужно иметь python версию фреймворка



# Keras

<https://keras.io/>

<https://keras.rstudio.com/>

<https://tensorflow.rstudio.com/keras/>

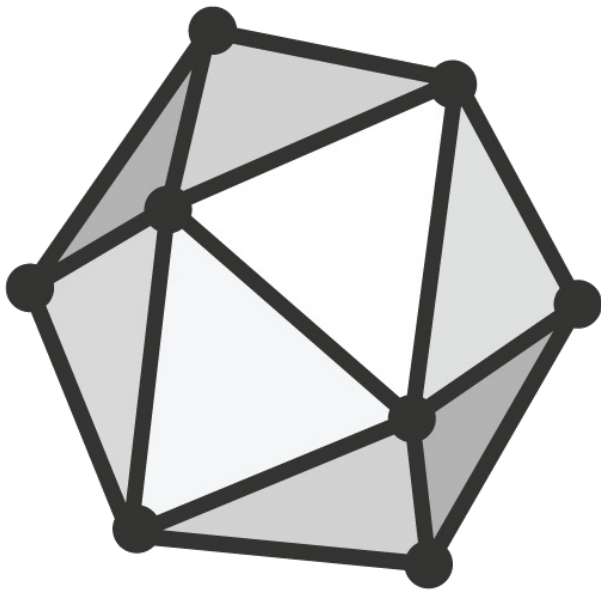
- Высокоуровневый фреймворк над другими такими бэкендами как Theano, CNTK, Tensorflow, и еще некоторые на подходе
- Делается Франсуа Шолле, который написал книгу Deep Learning in R
- Очень простой код
- Один и тот же код работает на разных бэкендах, что теоретически может быть полезно (нет)
- Есть очень много блоков нейросетей из современных SOTA работ
- Нивелирует боль статических вычислительных графов
- Уже давно дефолтом поставляется вместе с TensorFlow как его часть, но можно использовать и отдельно



<https://mxnet.apache.org/>

<https://github.com/apache/incubator-mxnet/tree/master/R-package>

- Является проектом Apache
- Сочетает в себе динамические и статические графы
- Тоже имеет зоопарк предобученных моделей
- Как и TensorFlow поддерживается многими языками, что может быть очень полезно
- Довольно разумный и хороший фреймворк, непонятно, почему не пользуется популярностью



# ONNX

<https://onnx.ai/>

<https://onnx.ai/onnx-r/>

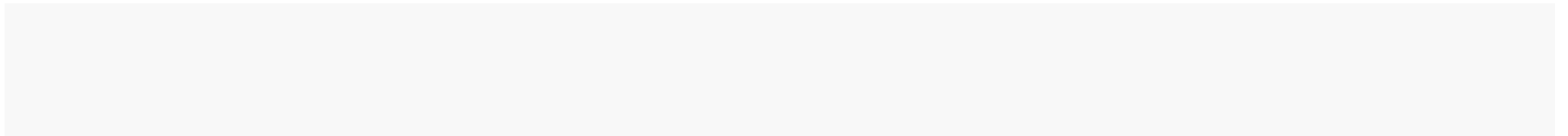
- Предоставляет открытый формат представления вычислительных графов, чтобы можно было обмениваться запуском одних и тех же, экспортированных в этот формат, модели с помощью разных фреймворков и своего рантайма
- Можно работать из R
- Изначально делался Microsoft вместе с Facebook
- Поддерживает кучу фреймворков нативно и конвертацию в ML и TF, Keras



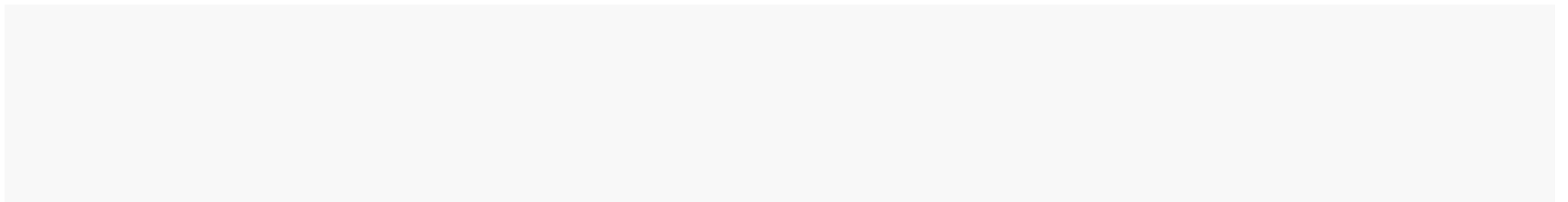
# Deep Learning with MXNet

# Установка

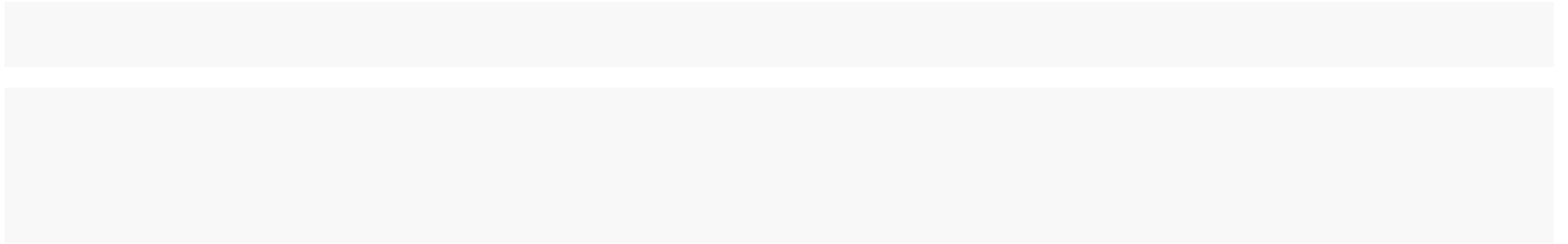
В Windows и MacOS в R



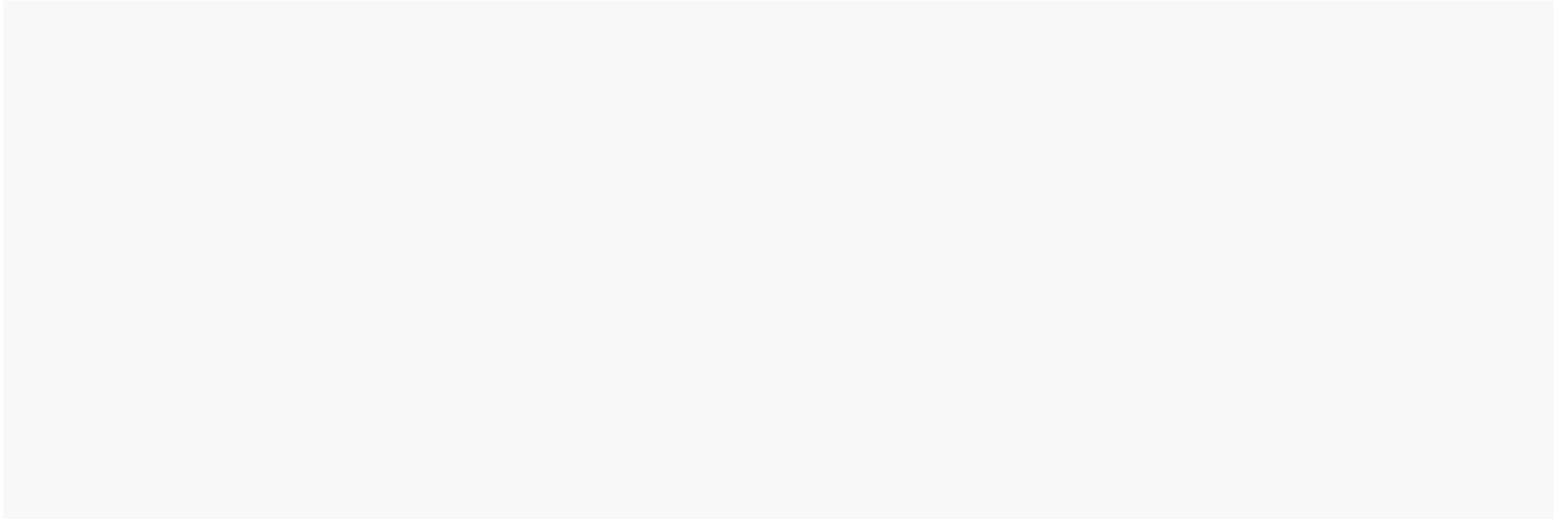
Linux bash



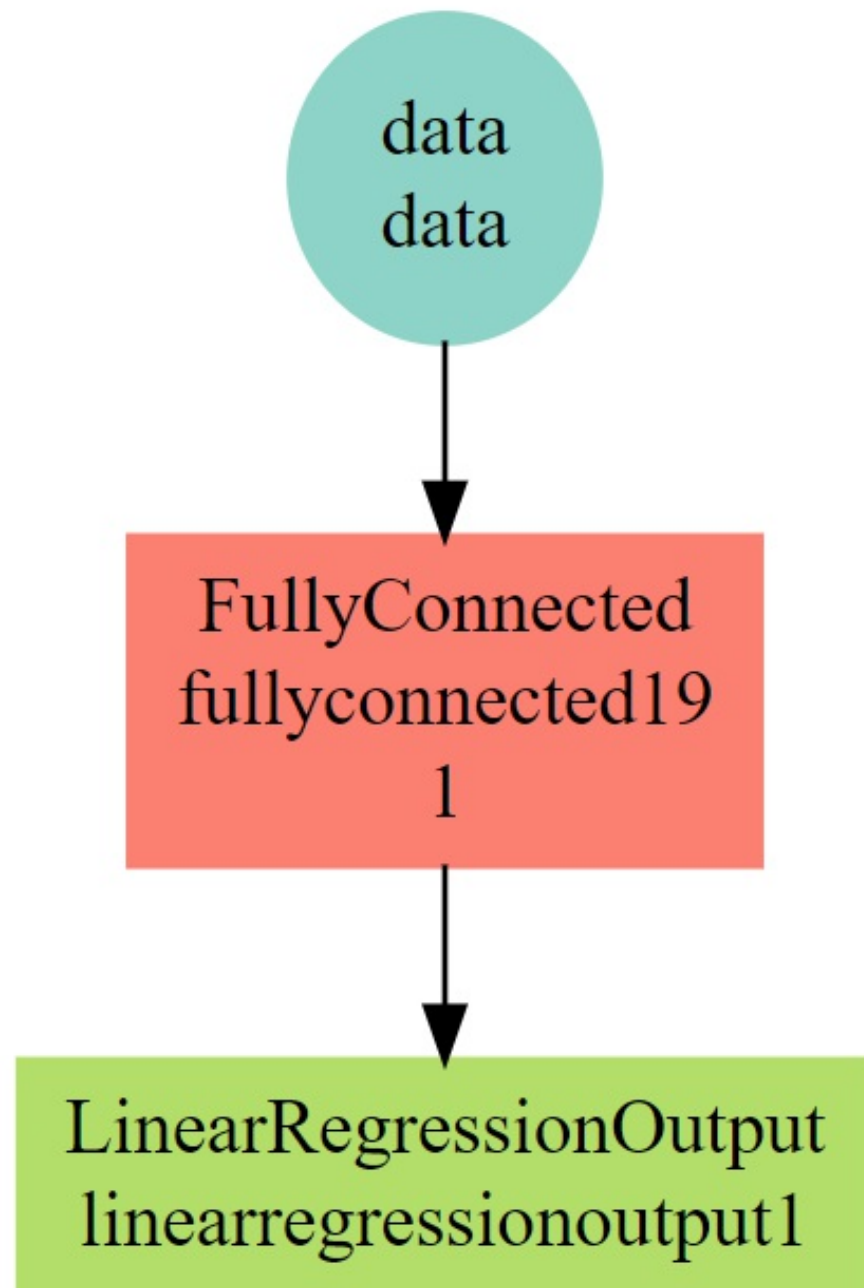
## Загрузка и обработка данных

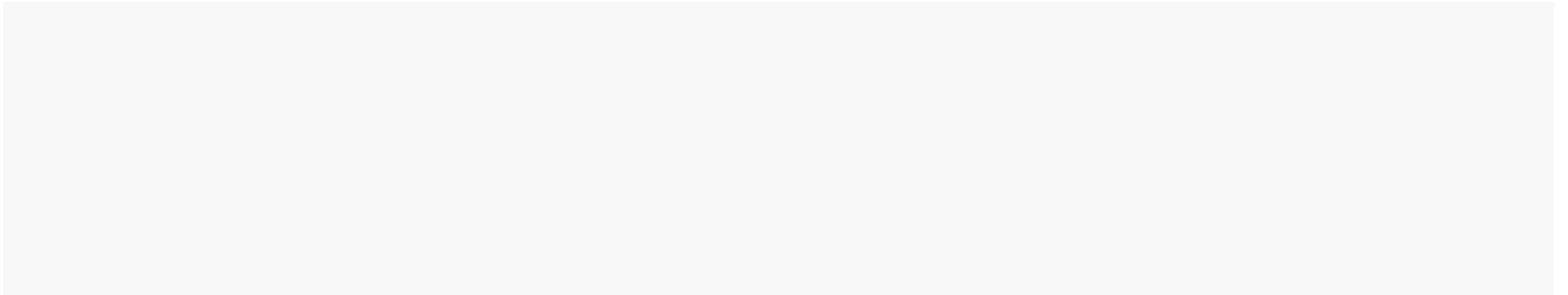


# Задания архитектуры сети

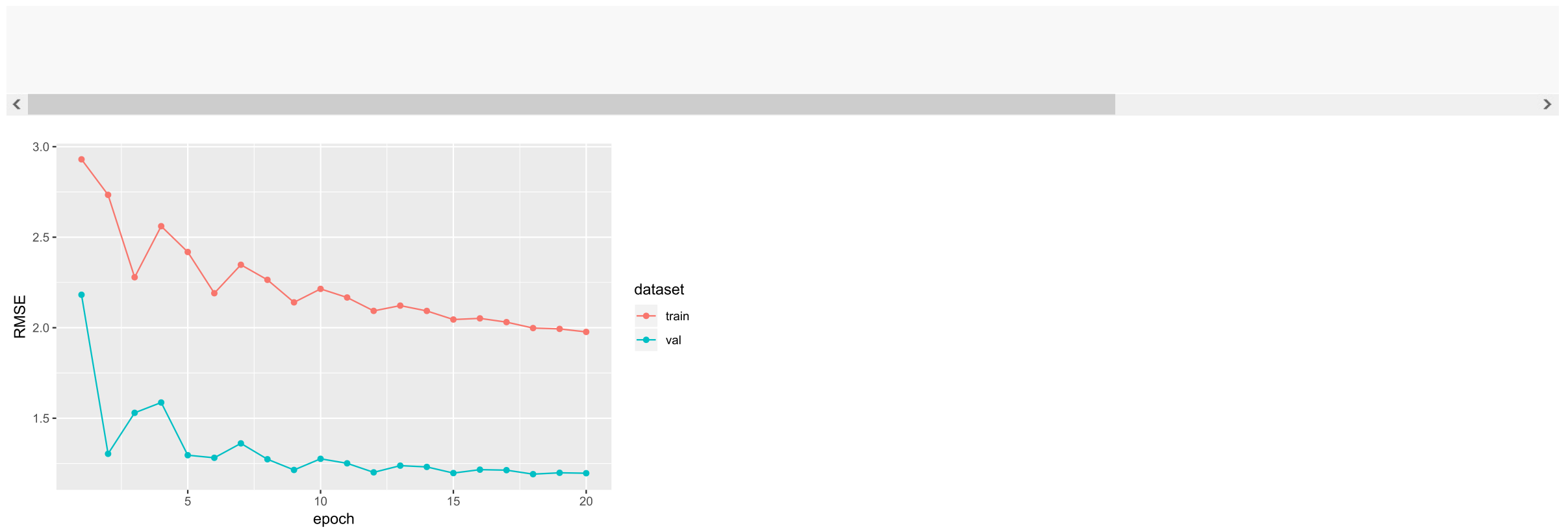


## Построим граф модели





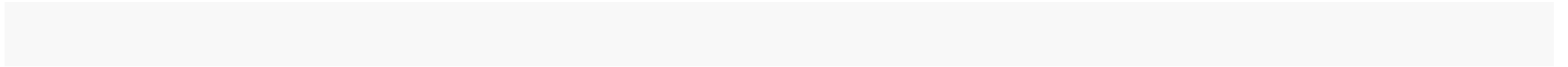
# Построим кривую обучения



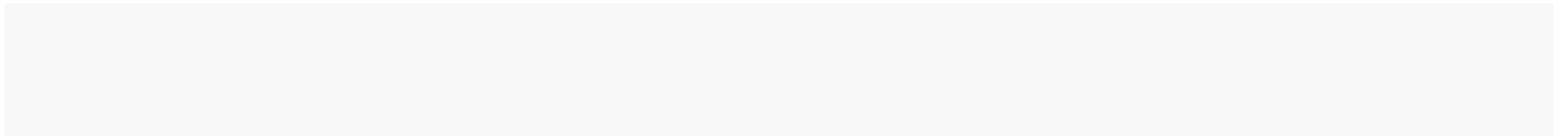
# Deep Learning with Keras



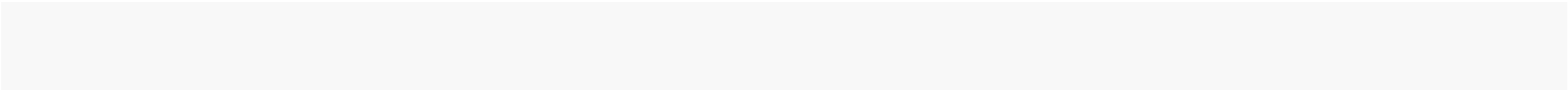
## Установка



## Загрузка нужных нам пакетов

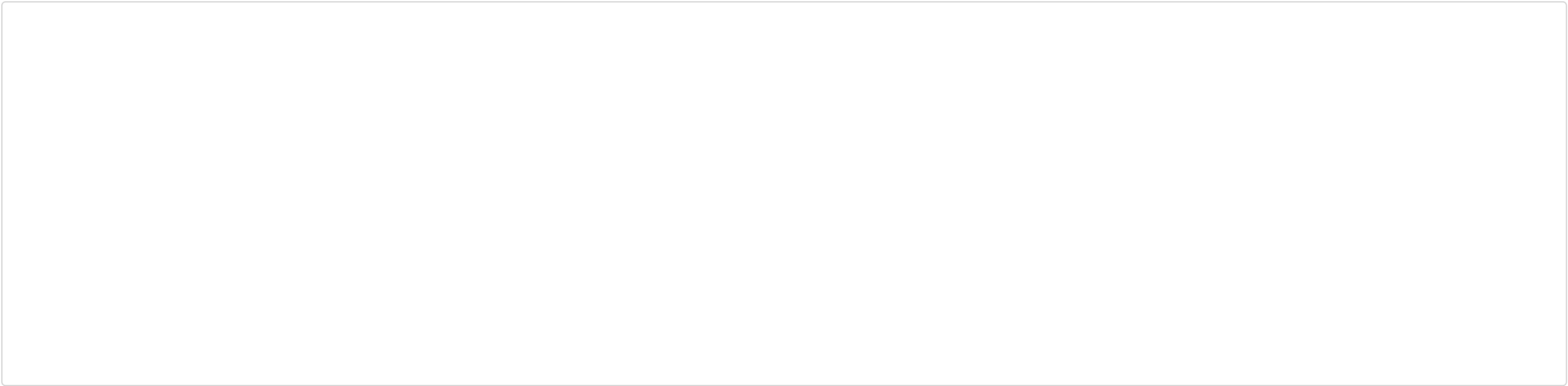
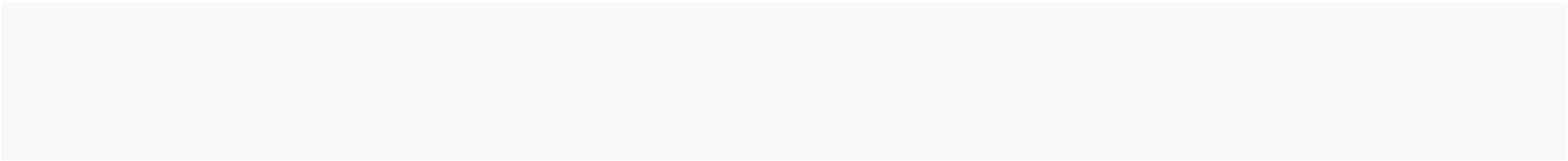


# Загрузка данных

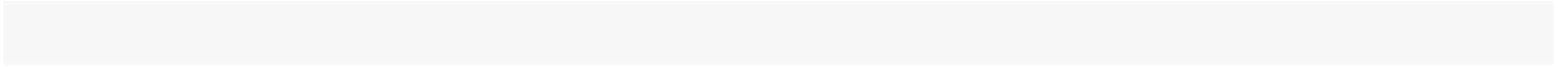


number label

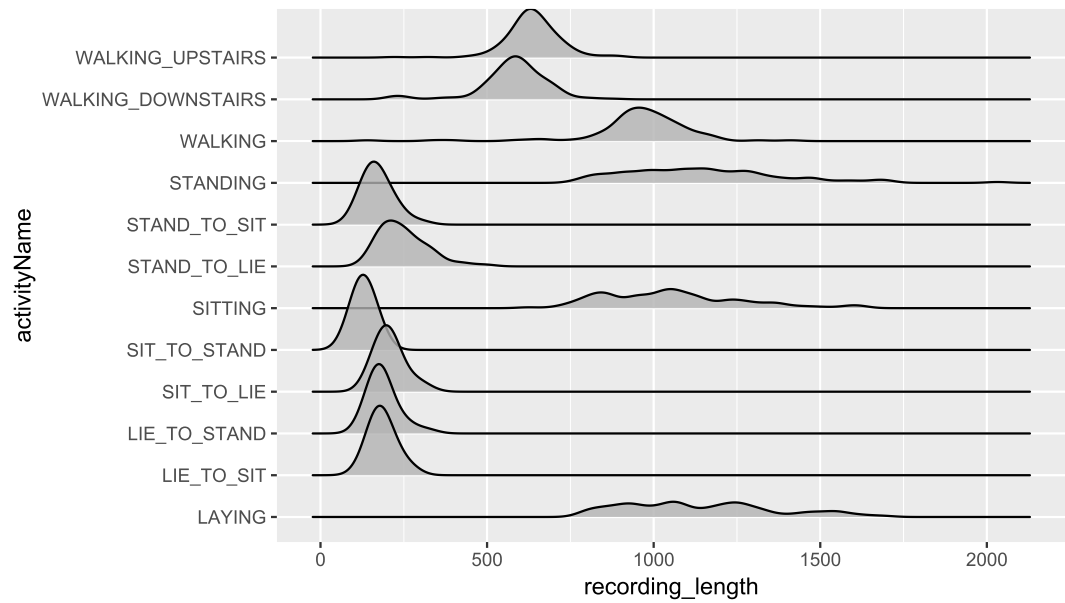
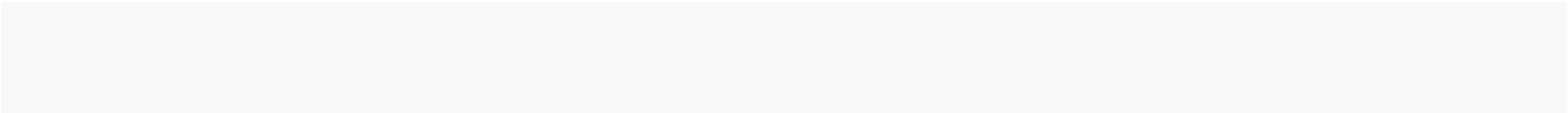
- 
- 1 WALKING
  - 2 WALKING\_UPSTAIRS
  - 3 WALKING\_DOWNSTAIRS 4 SITTING
  - 5 STANDING
  - 6 LAYING
  - 7 STAND\_TO\_SIT
  - 8 SIT\_TO\_STAND
  - 9 SIT\_TO\_LIE
  - 10 LIE\_TO\_SIT
  - 11 STAND\_TO\_LIE
  - 12 LIE\_TO\_STAND



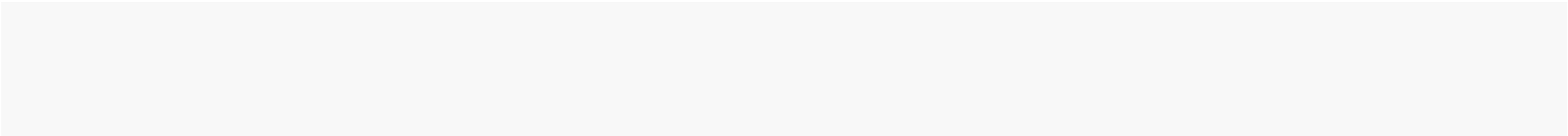
TLDR



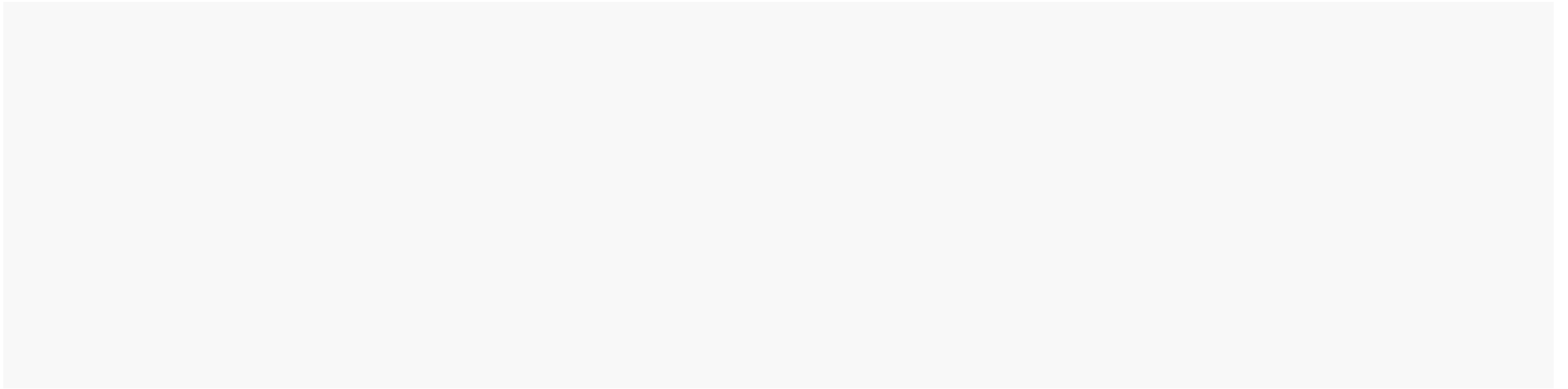
# Посмотрим на данные



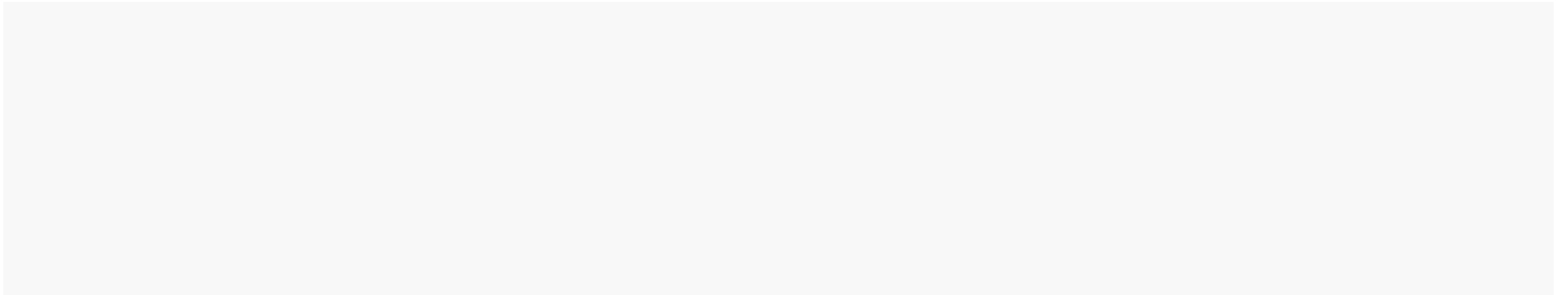
# Отфильтруем



Разделим на трейн тест

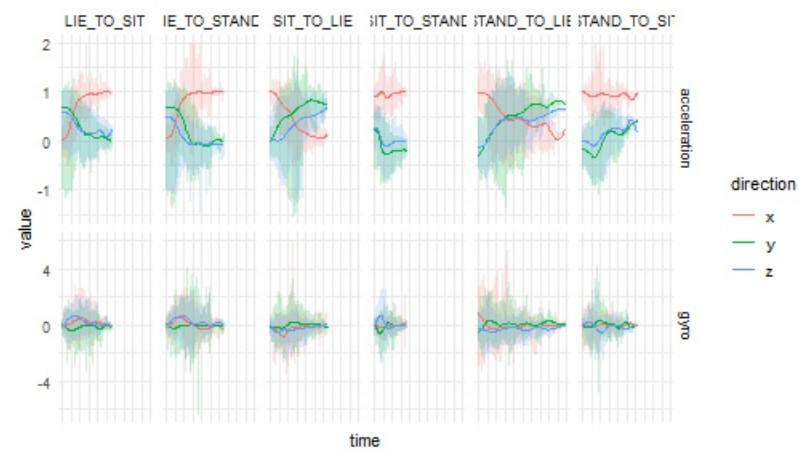


## Посмотрим собственно на активности по классам

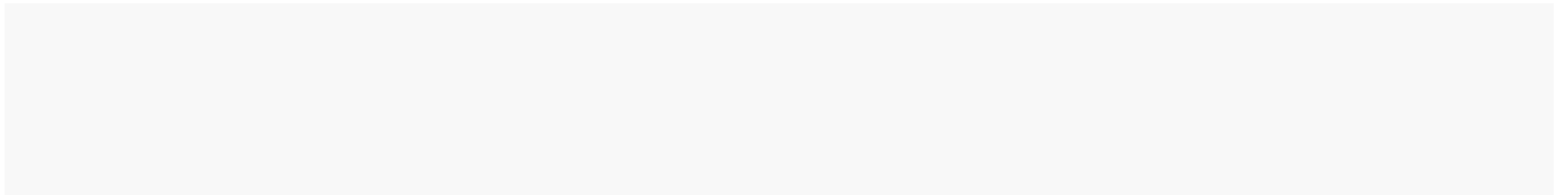
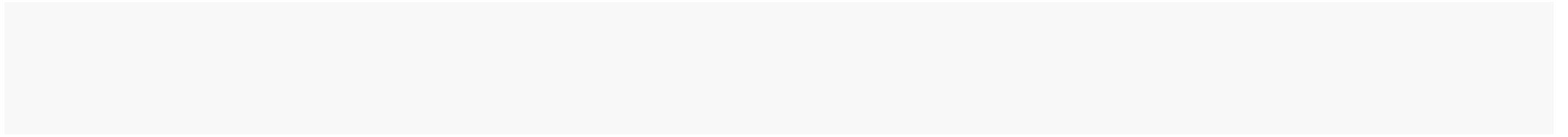




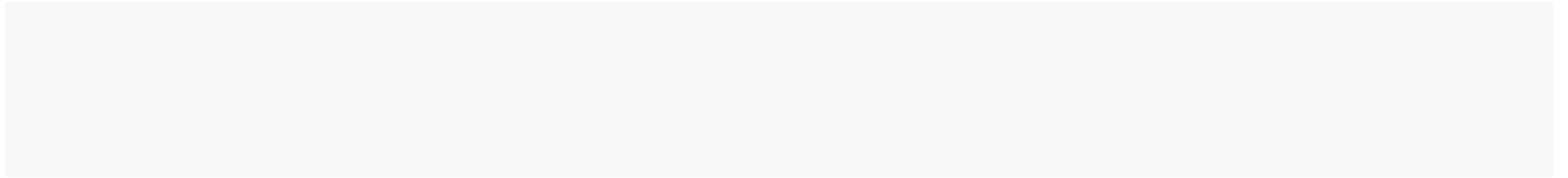
# Посмотрим собственно на активности по классам



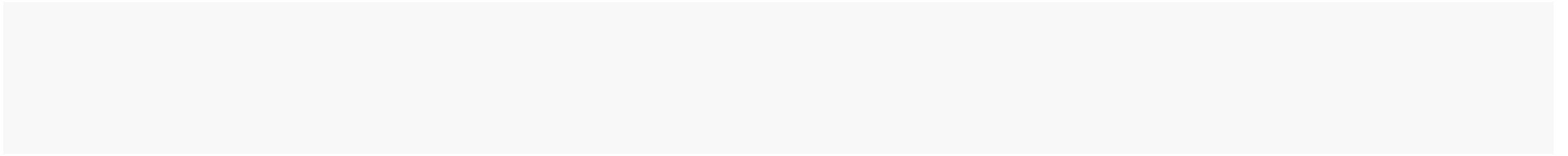
## Подготовка данных к обучению



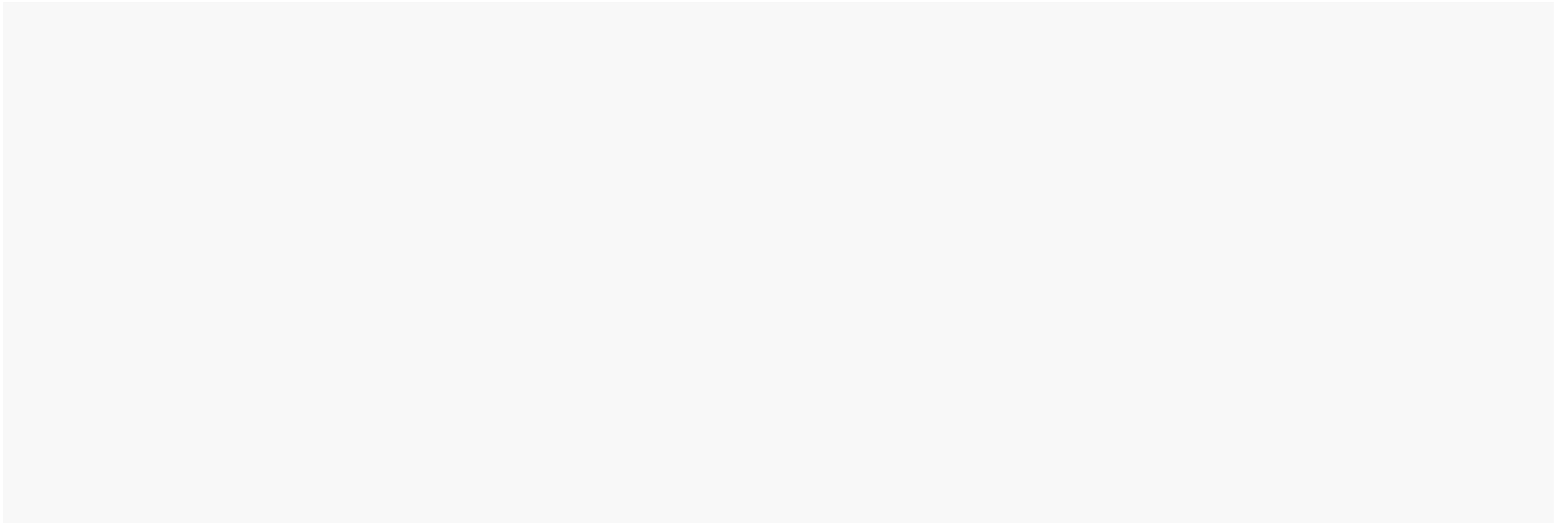
# Подготовка данных к обучению



Наконец то сетка!



Наконец то сетка!



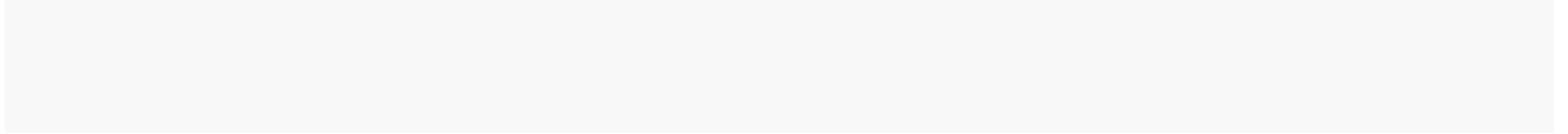
# Наконец то сетка!

Выведем описание нашей сетки

Layer (type)	Output Shape	Param #
conv1d_7 (Conv1D)	(None, 327, 24)	1176
batch_normalization_10 (BatchNormalization)	(None, 327, 24)	96
spatial_dropout1d_4 (SpatialDropout1D)	(None, 327, 24)	0
conv1d_8 (Conv1D)	(None, 320, 12)	2316
global_average_pooling1d_4 (GlobalAveragePooling1D)	(None, 12)	0
batch_normalization_11 (BatchNormalization)	(None, 12)	48
dropout_7 (Dropout)	(None, 12)	0
dense_4 (Dense)	(None, 48)	624
batch_normalization_12 (BatchNormalization)	(None, 48)	192
dropout_8 (Dropout)	(None, 48)	0
dense_output (Dense)	(None, 6)	294
Total params: 4,746		
Trainable params: 4,578		
Non-trainable params: 168		

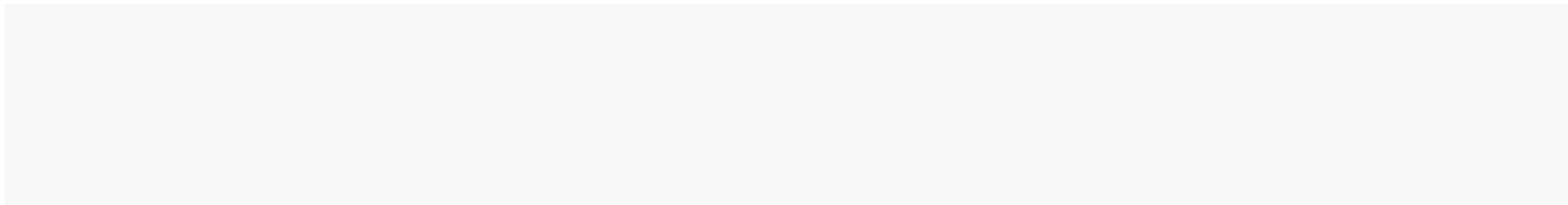
Обучим же наконец

Компиляция графа



Обучим же наконец

train

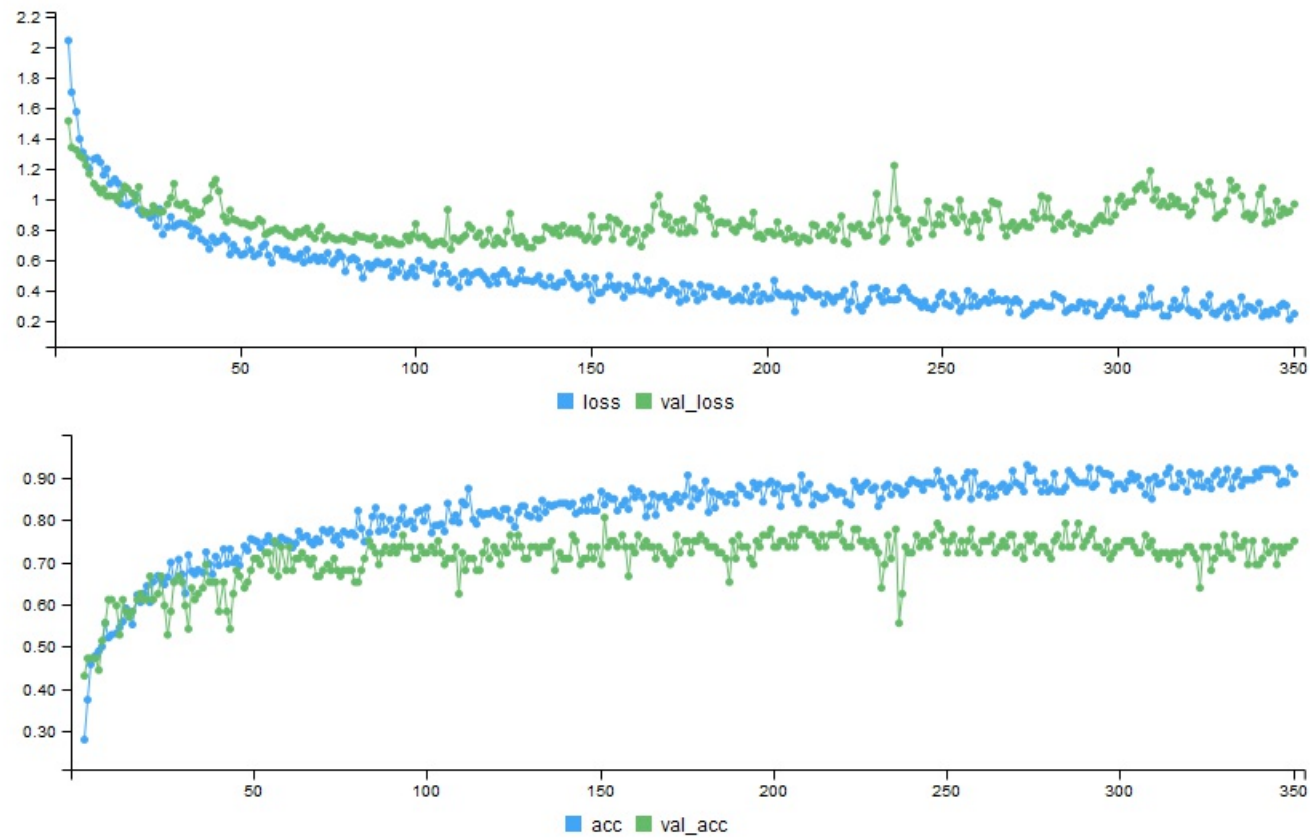




# Обучим же наконец

```
Train on 286 samples, validate on 72 samples
Epoch 1/350
286/286 [=====] - 3s 10ms/step - loss: 2.0388 - acc: 0.2797 - val_loss: 1.5122 - val_acc: 0.4306
Epoch 2/350
286/286 [=====] - 0s 483us/step - loss: 1.6998 - acc: 0.3741 - val_loss: 1.3388 - val_acc: 0.4722
Epoch 3/350
286/286 [=====] - 0s 500us/step - loss: 1.5719 - acc: 0.4580 - val_loss: 1.3223 - val_acc: 0.4722
Epoch 4/350
286/286 [=====] - 0s 531us/step - loss: 1.3942 - acc: 0.4790 - val_loss: 1.2848 - val_acc: 0.4722
Epoch 5/350
286/286 [=====] - 0s 610us/step - loss: 1.3068 - acc: 0.4895 - val_loss: 1.2697 - val_acc: 0.4444
Epoch 6/350
286/286 [=====] - 0s 486us/step - loss: 1.2666 - acc: 0.5000 - val_loss: 1.2210 - val_acc: 0.5139
Epoch 7/350
286/286 [=====] - 0s 580us/step - loss: 1.2000 - acc: 0.5559 - val_loss: 1.1661 - val_acc: 0.5556
Epoch 8/350
286/286 [=====] - 0s 503us/step - loss: 1.2626 - acc: 0.5210 - val_loss: 1.1003 - val_acc: 0.6111
Epoch 9/350
286/286 [=====] - 0s 514us/step - loss: 1.2714 - acc: 0.5280 - val_loss: 1.0738 - val_acc: 0.6111
```

# Обучим же наконец



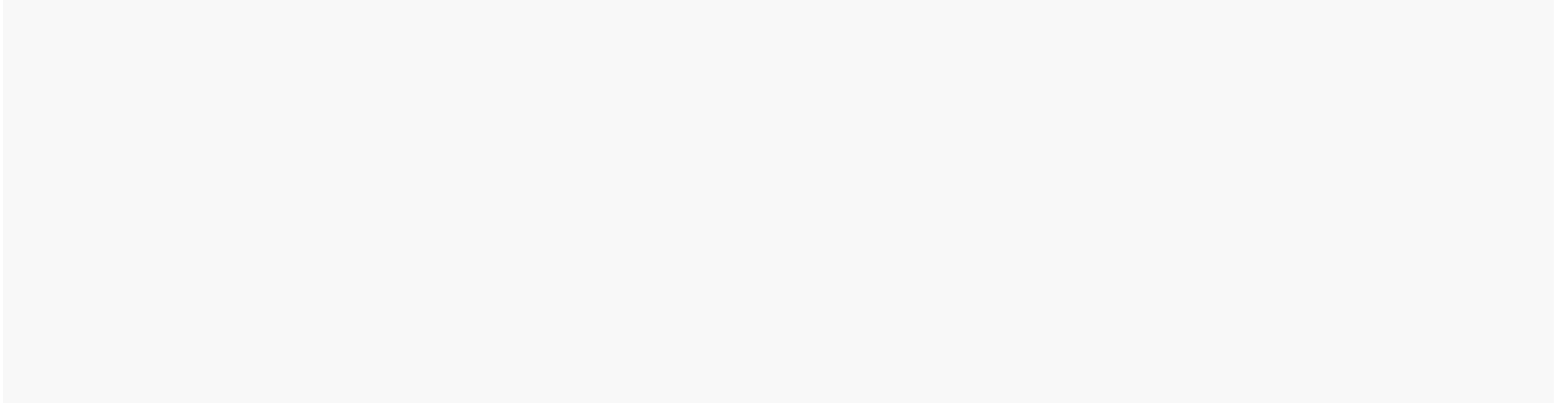
Предсказание

Подготовка теста

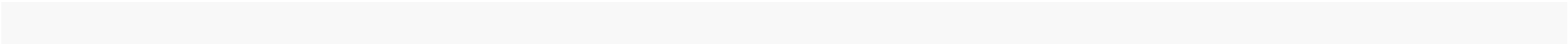
Выбор лучшей модели

Предсказание

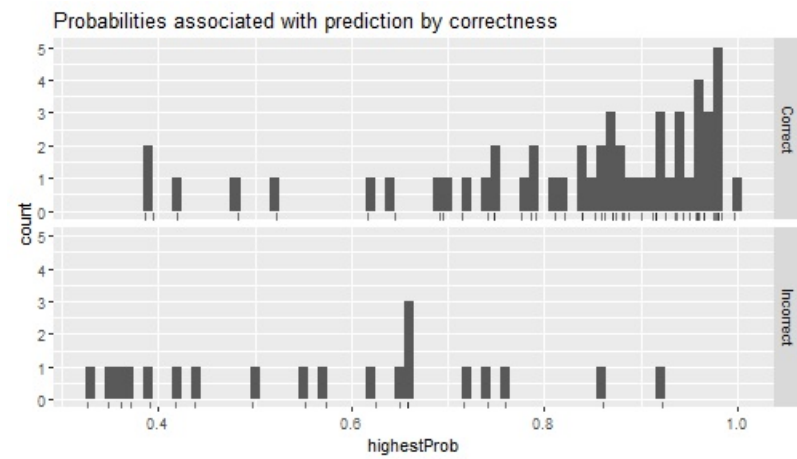
Еще немного кода



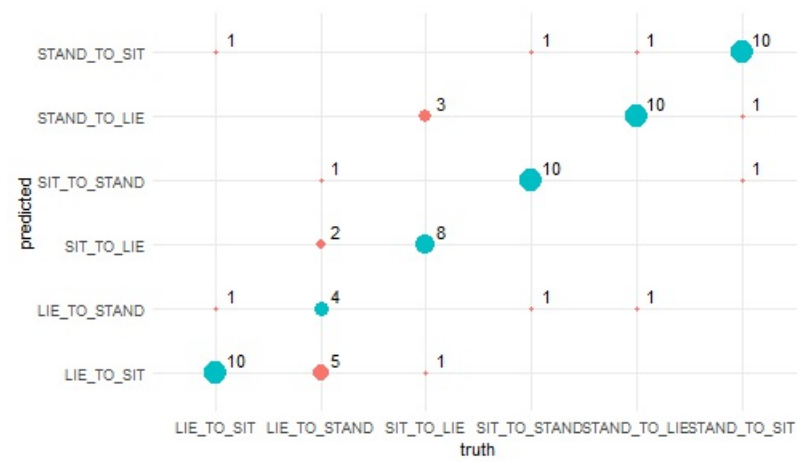
# Предсказание



# Визуализация ошибок

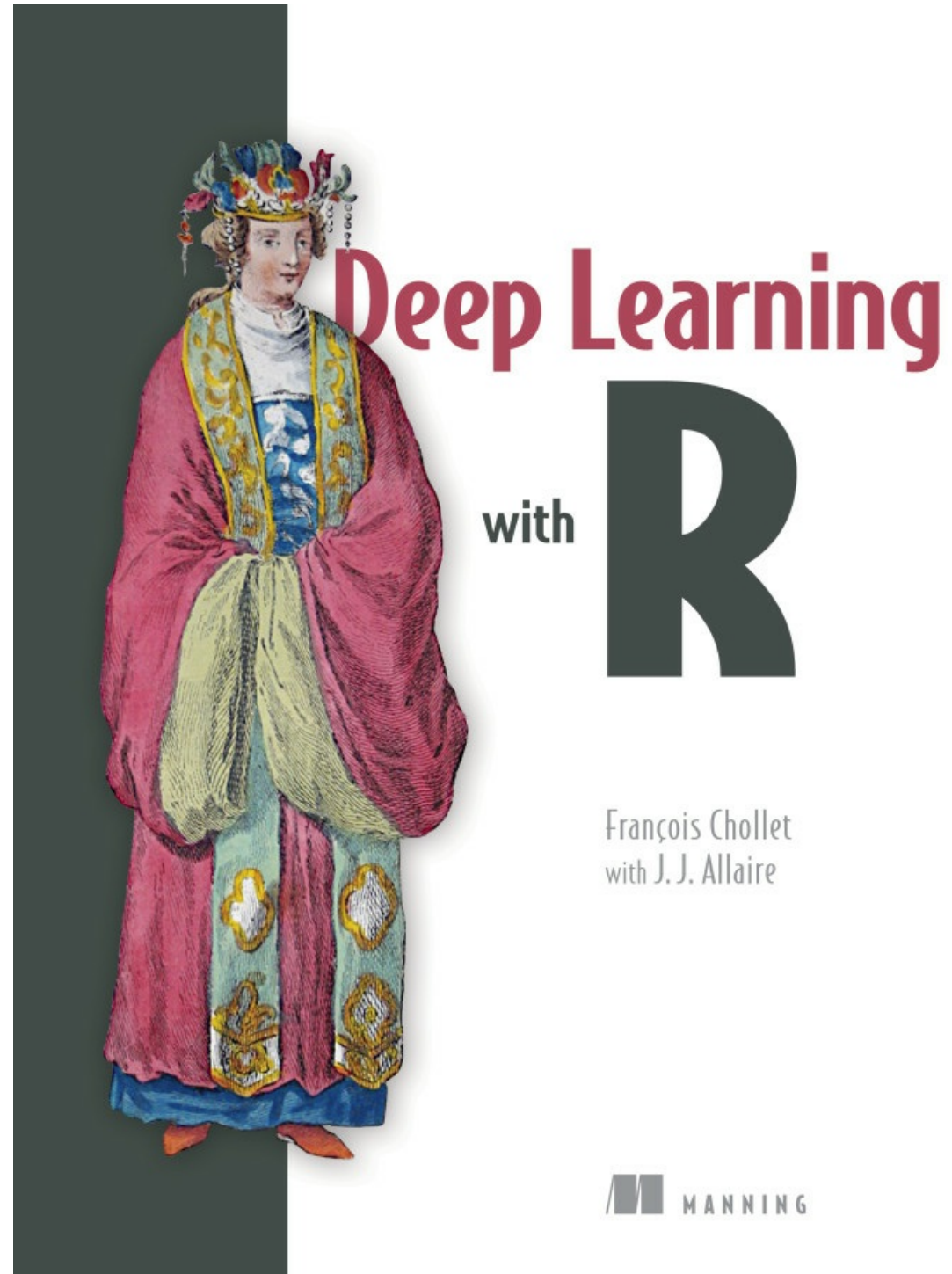


# Визуализация ошибок



## Заклучение





# Спасибо!

Слайды сделаны с помощью R package **xaringan**.

Веб версию слайдов можно найти на [https://metya.github.io/DeepLearning\\_n\\_R/](https://metya.github.io/DeepLearning_n_R/)

Код можно посмотреть здесь [https://github.com/metya/DeepLearning\\_n\\_R/](https://github.com/metya/DeepLearning_n_R/)