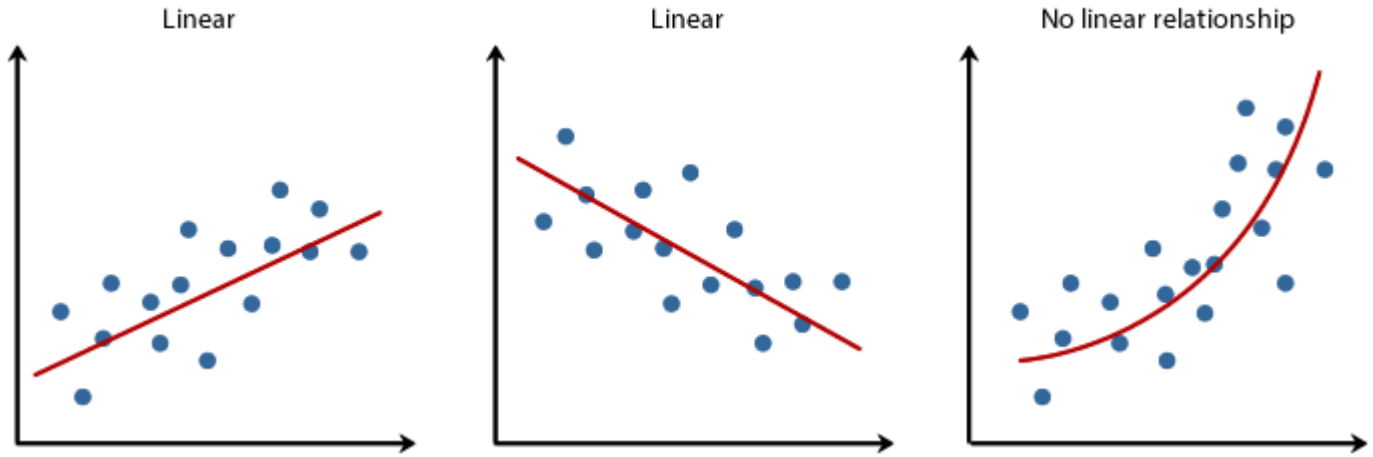

بسم الله الرحمن الرحيم

تابع ضرر و تابع فعال‌سازی

رگرسیون

- شامل تخمین یک مقدار پیوسته است
  - به عنوان مثال، پیش‌بینی دمای فردا بر اساس داده‌های هواشناسی
- می‌توانیم از تابع فعال‌سازی خطی در لایه آخر استفاده کنیم
  - تابع ضرر مناسب چیست؟
  - با فرض توزیع نرمال خطا، میانگین مربعات خطا



$$J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{x},\boldsymbol{y}\sim\hat{p}_{data}} \|\boldsymbol{y} - f(\boldsymbol{x};\boldsymbol{\theta})\|^2$$

تخمین قیمت خانه

- مجموعه داده Boston Housing Price
  - نسبتا کوچک: ۴۰۴ داده آموزشی، ۱۰۲ داده آزمون
  - ویژگی‌های مجموعه داده دارای مقیاس‌های متفاوتی هستند
  - قیمت‌ها برحسب هزار دلار هستند

```
from keras.datasets import boston_housing
(train_data, train_targets), (test_data, test_targets) = boston_housing.load_data()
```

```
print(train_data.shape)
print(train_targets.shape)
```

```
(404, 13)
(404,)
```

```
std = train_data.std(axis=0)
print(std)
```

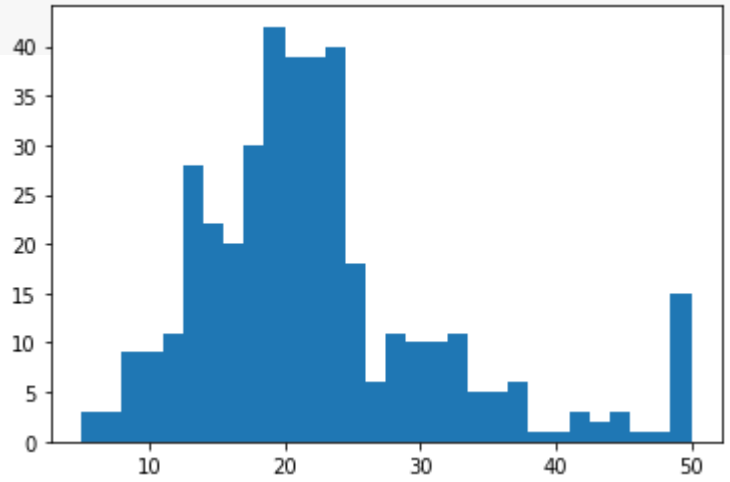
```
[9.22929073e+00 2.37382770e+01 6.80287253e+00 2.40939633e-01
 1.17147847e-01 7.08908627e-01 2.79060634e+01 2.02770050e+00
 8.68758849e+00 1.66168506e+02 2.19765689e+00 9.39946015e+01
 7.24556085e+00]
```

تخمین قیمت خانه

- مجموعه داده Boston Housing Price
  - نسبتا کوچک: ۴۰۴ داده آموزشی، ۱۰۲ داده آزمون
  - ویژگی‌های مجموعه داده دارای مقیاس‌های متفاوتی هستند
  - قیمت‌ها برحسب هزار دلار هستند

```
print(train_targets)
plt.hist(train_targets, 30)
```

```
[ 15.2, 42.3, 50. ... 19.4, 19.4, 29.1]
```



تخمین قیمت خانه

- داشتن ورودی‌هایی که مقیاس‌ها و بازه‌های متفاوتی دارند آموزش مدل را با چالش مواجه می‌کند
- راه حل: نرمال‌سازی ویژگی‌ها
  - میانگین نمونه‌های آموزشی را کم و بر انحراف معیار آنها تقسیم می‌کنیم
  - میانگین ویژگی‌ها ۰ و انحراف معیار آنها ۱ خواهد شد

```
mean = train_data.mean(axis=0)
std = train_data.std(axis=0)
train_data -= mean
train_data /= std
test_data -= mean
test_data /= std
```

تخمین قیمت خانه

- مجموعه داده کوچک، مدل کوچک

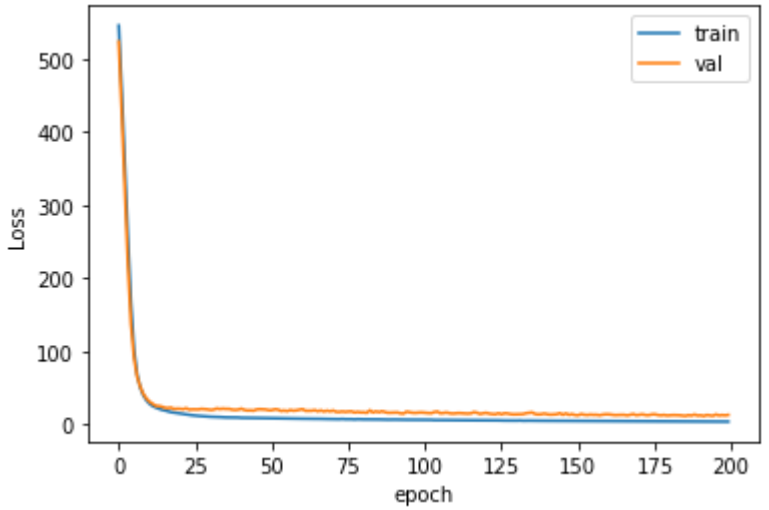
```
model = keras.models.Sequential()
model.add(keras.layers.Input(train_data.shape[1]))
model.add(keras.layers.Dense(64, activation='relu'))
model.add(keras.layers.Dense(64, activation='relu'))
model.add(keras.layers.Dense(1))
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
model.summary()
```

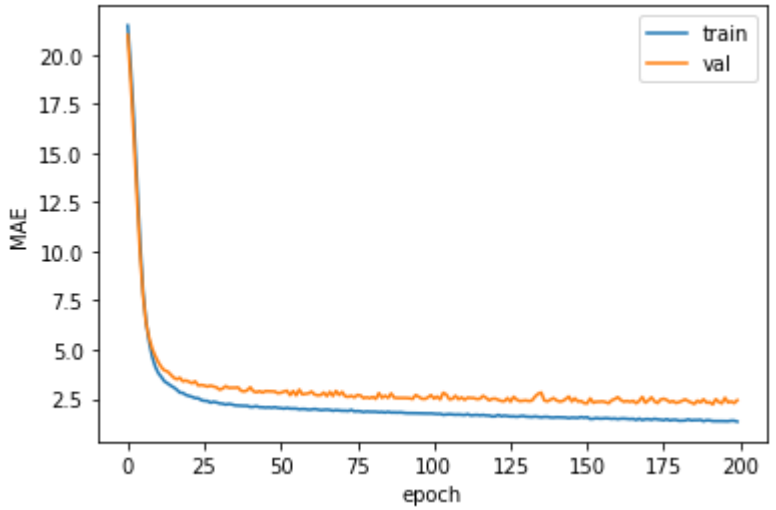
```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_3 (Dense)              (None, 64)                896
_________________________________________________________________
dense_4 (Dense)              (None, 64)                4160
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 65
=================================================================
Total params: 5,121
Trainable params: 5,121
Non-trainable params: 0
```

تخمین قیمت خانه

- مجموعه داده کوچک، مدل کوچک

```
history = model.fit(train_data, train_targets,
                    validation_data=(test_data, test_targets),
                    batch_size=32,
                    epochs=200)
```






```
Epoch 200/200
13/13 [==============================] - 0s 5ms/step - loss: 4.4936 - mae: 1.4073 - val_loss: 12.7377 - val_mae: 2.4284
```

https://colab.research.google.com/drive/1Rj2owHgHtI5HqC15WkmeR2WozXbOhLV0?usp=sharing

دسته‌بندی باینری

- در بسیاری از مسائل یادگیری ماشین نیاز به پیش‌بینی مقدار یک متغیر باینری است
- شبکه‌های عصبی باید تنها یک مقدار را پیش‌بینی کنند $P(y = 1|\boldsymbol{x}) \in [0,1]$
- بهتر است خروجی محدود به بازه $[0,1]$ باشد

	Two Class Classification	
$y \in \{0,1\}$	**1 or Positive Class**	**0 or Negative Class**
Email	Spam	Not Spam
Tumor	Malignant	Benign
Transaction	Fraudulent	Not Fraudulent

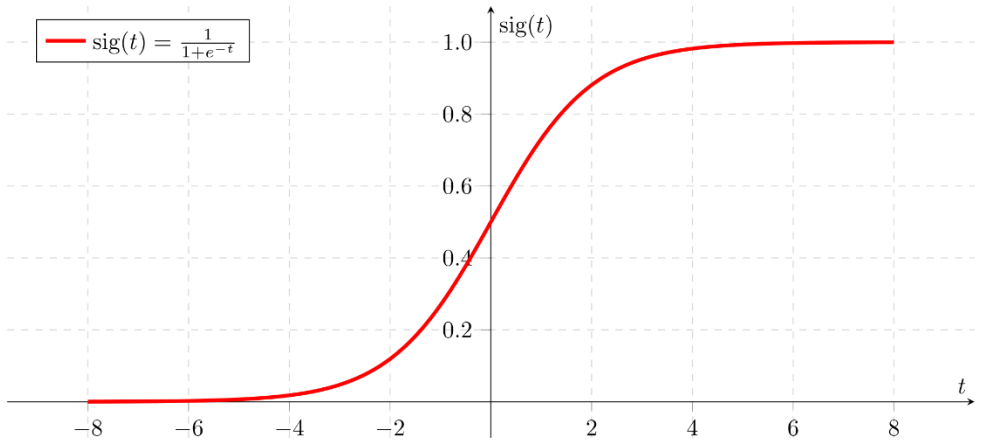
e.g. $P(y = 1|\boldsymbol{x}) = \max\{0, \min\{1, \boldsymbol{w}^T \boldsymbol{h} + b\}\}$

- بهینه‌سازهای مبتنی بر گرادیان نمی‌توانند پارامترهای مطلوب چنین شبکه‌ای را بیابند
  - مشتق تابع ضرر برای داده‌هایی که کاملا اشتباه پیش‌بینی شوند ۰ است

دسته‌بندی باینری

- بهتر است از تابعی برای محدود کردن خروجی استفاده کنیم که مشتق آن صفر نشود
- تابع سیگموئید برای این منظور پرکاربرد است $P(y=1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{h} + b)$
- آیا تابع ضرر MSE در این حالت مناسب است؟

$$J(\boldsymbol{\theta}) = \big(y - \sigma(z)\big)^2, \qquad z = \boldsymbol{w}^T\boldsymbol{h} + b$$



- مثال عددی: اگر $z = 5.6$ و $y = 0$

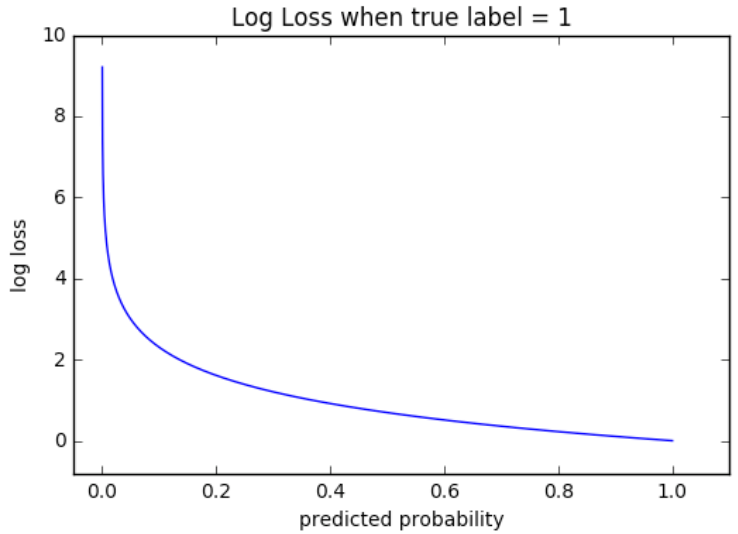
$$\frac{dJ}{dz} = -2\big(y - \sigma(z)\big)\sigma(z)\big(1 - \sigma(z)\big) = 0.0073$$

$$\approx -2(0-1)1(1-1)$$

دسته‌بندی باینری

- بهتر است از تابعی برای محدود کردن خروجی استفاده کنیم که مشتق آن صفر نشود
- تابع سیگموئید برای این منظور پرکاربرد است $P(y=1|\boldsymbol{x}) = \sigma(\boldsymbol{w}^T\boldsymbol{h} + b)$
- با استفاده از ML تابع ضرر binary cross-entropy بدست می‌آید

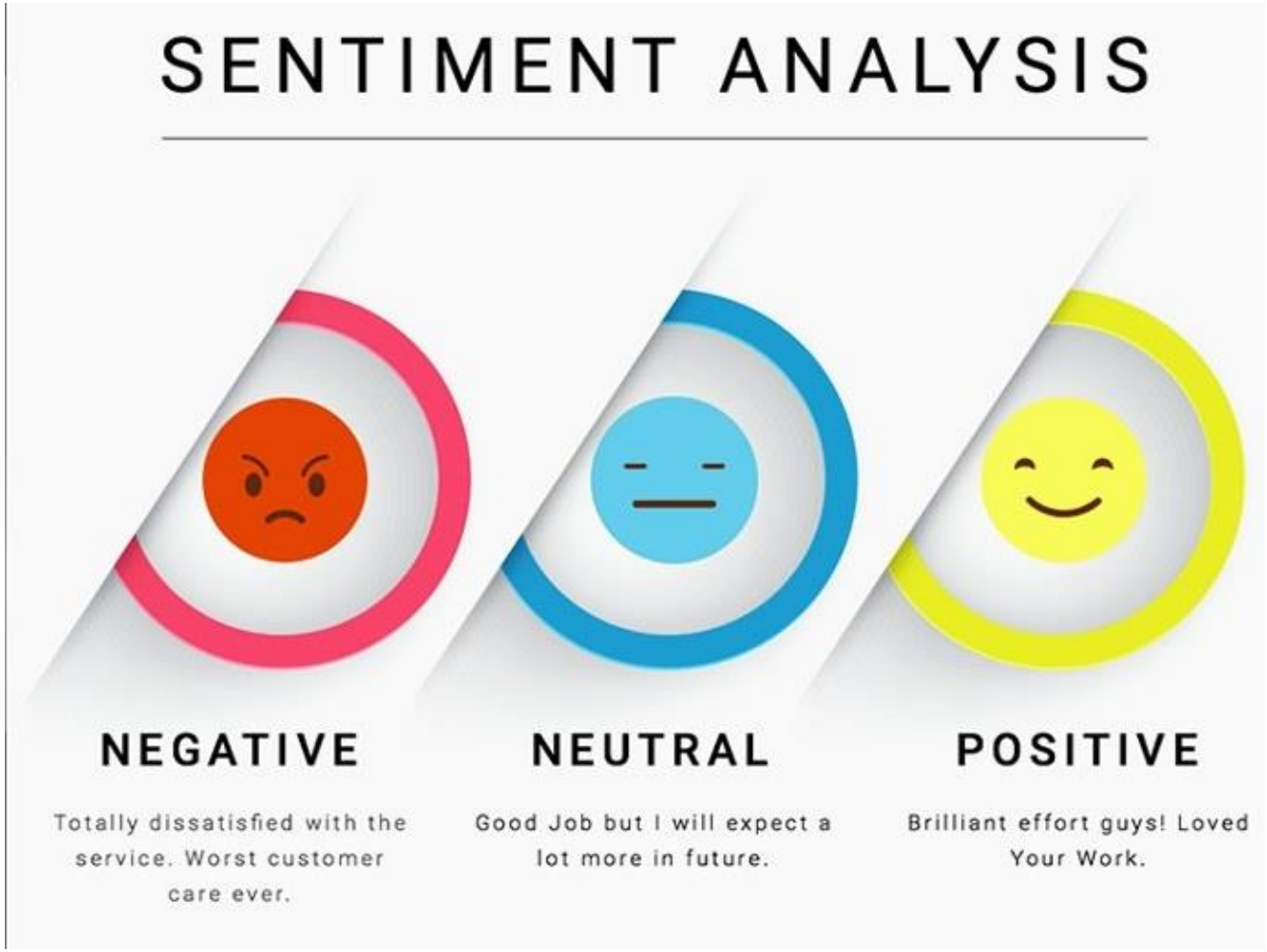
$$J(\boldsymbol{\theta}) = -y \log \sigma(z) - (1-y) \log\big(1 - \sigma(z)\big)$$



- مثال عددی: اگر $z = 5.6$ و $y = 0$

$$\frac{dJ}{dz} = -\frac{-\sigma(z)\big(1-\sigma(z)\big)}{1-\sigma(z)} = \sigma(z) = 0.9963$$

دسته‌بندی نظرها



- مجموعه داده IMDB
  - ۵۰،۰۰۰ نظر بسیار قطبی
  - ۲۵،۰۰۰ برای آموزش و ۲۵،۰۰۰ برای آزمون
  - ۵۰٪ نظرها مثبت و ۵۰٪ منفی
  - هر نظر یک دنباله از کلمات است که به دنباله‌ای از اعداد صحیح تبدیل می‌شود
  - هر عدد صحیح متناظر با کلمه خاصی در فرهنگ لغت است

دسته‌بندی نظرها

```
from keras.datasets import imdb
(train_data,train_labels), (test_data,test_labels) = imdb.load_data(num_words=1000)
```

```
d_view = [(v, k) for k, v in imdb.get_word_index().items()]
d_view.sort()
for v, k in d_view[:20]:
    print("{}:\t{}".format(k, v))
for v, k in d_view[-20:]:
    print("{}:\t{}".format(k, v))
```

```
the:    1
and:    2
a:      3
of:     4
to:     5
is:     6
br:     7
in:     8
it:     9
i:      10
this:   11
that:   12
was:    13
as:     14
for:    15
with:   16
movie:  17
but:    18
film:   19
on:     20
percival:       88565
lubricated:     88566
heralding:      88567
baywatch':      88568
odilon: 88569
'solve':        88570
guard's:        88571
nemesis':       88572
airsoft:        88573
urrrghhh:       88574
ev:     88575
chicatillo:     88576
transacting:    88577
sics:   88578
wheelers:       88579
pipe's: 88580
copywrite:      88581
artbox: 88582
voorhees':      88583
'l':    88584
```

```
<START> this film was just brilliant casting location scenery story direction

1, 14, 22, 16, 43, 530, 973, 1622, 1385, 65, 458, 4468, …

1, 14, 22, 16, 43, 530, 973,    2,    2, 65, 458,    2, …
```

دسته‌بندی نظرها

```
from keras.datasets import imdb
(train_data,train_labels), (test_data,test_labels) = imdb.load_data(num_words=1000)
```

- بردارهای train_labels و test_labels شامل مقادیر ۰ (نظر منفی) و ۱ (نظر مثبت) هستند
- هر کدام از نظرها یک بردار از اعداد صحیح است که طول آنها متفاوت است

```
np.array([len(d) for d in train_data])
```

```
array([218, 189, 141, ..., 184, 150, 153])
```

- می‌توانیم بردارها را pad کنیم تا طول آنها یکسان شود
- یک کدینگ ساده به این صورت است که یک بردار با طول تعداد کلمات بسازیم که اندیس کلمات موجود در جمله برابر با ۱ و باقی مقادیر ۰ باشند

دسته‌بندی نظرها

```
def vectorize_sequences(sequences, dimension=10000):
    results = np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence] = 1.
    return results

x_train = vectorize_sequences(train_data)
x_test = vectorize_sequences(test_data)
```

```
print(x_train.shape)
print(x_train[0])
```

```
(25000, 10000)
[0. 1. 1. ... 0. 0. 0.]
```

```
y_train = np.asarray(train_labels).astype('float32')
y_test = np.asarray(test_labels).astype('float32')
```