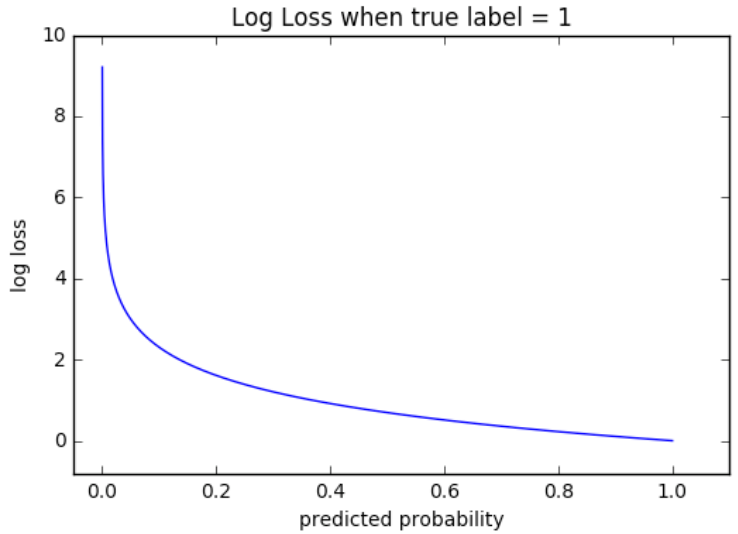

بسم الله الرحمن الرحيم

# تابع ضرر و تابع فعال‌سازی

# 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

```
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])
```

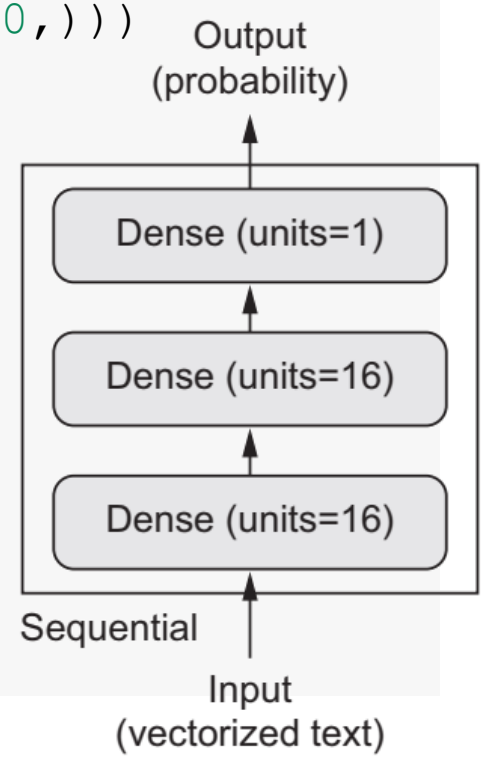
```
(25000, 10000)
[0. 1. 1. ... 0. 0. 0.]
```

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

# دسته‌بندی نظرها

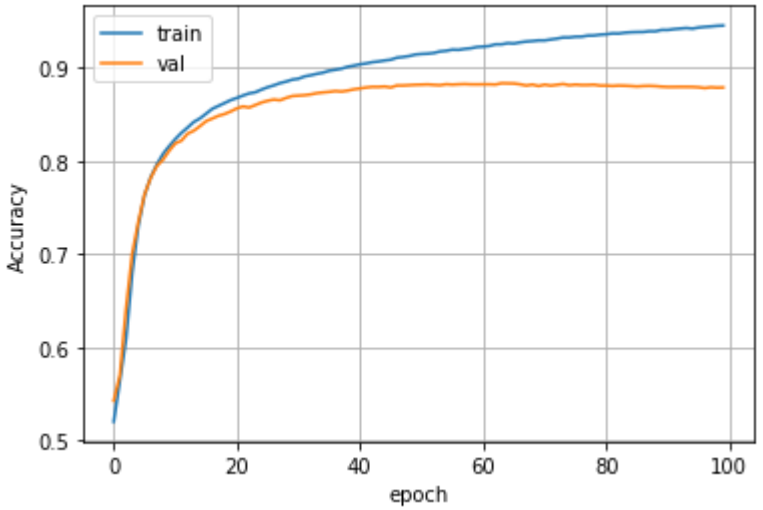
- یک شبکه ۳ لایه ساده

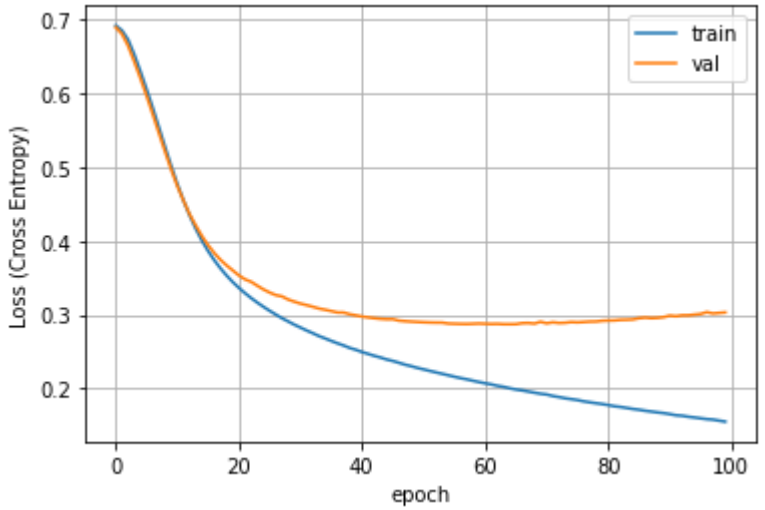
```
model = keras.models.Sequential()
model.add(keras.layers.Dense(16, activation='relu', input_shape=(10000,)))
model.add(keras.layers.Dense(16, activation='relu'))
model.add(keras.layers.Dense(1, activation='sigmoid'))

model.compile(optimizer='sgd',
              loss='binary_crossentropy',
              metrics=['accuracy'])

history = model.fit(x_train, y_train,
                    validation_data=(x_val, y_val),
                    epochs=100,
                    batch_size=512)
```

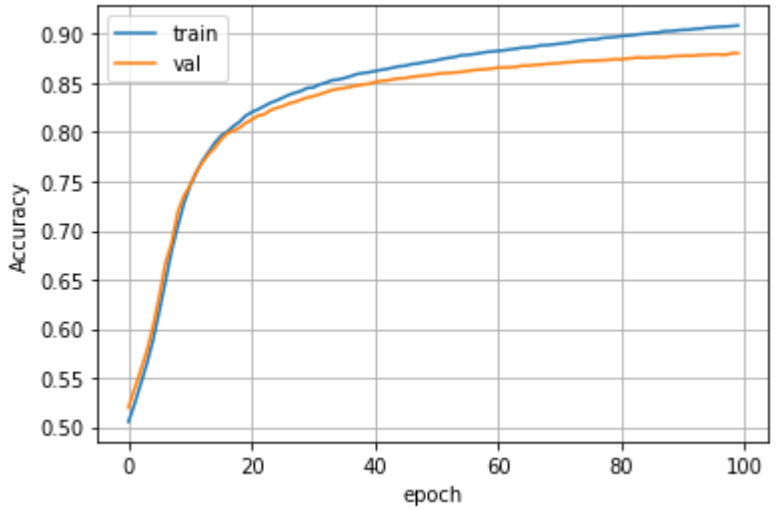


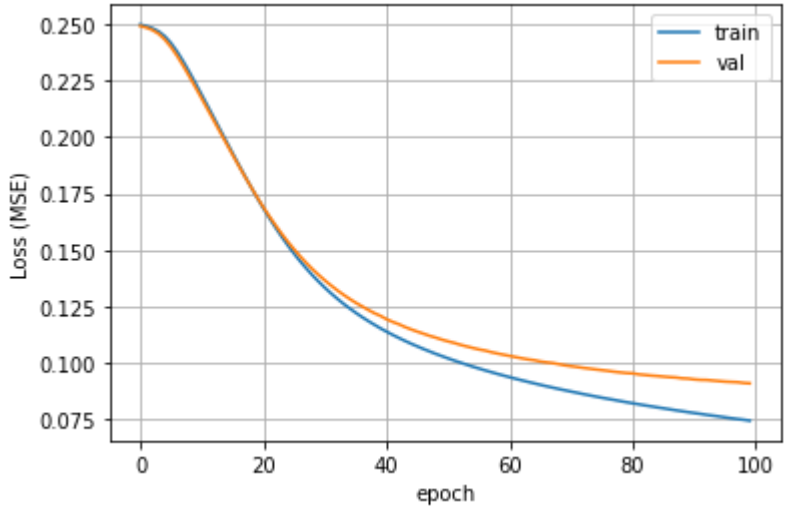
# دسته‌بندی نظرها

- بهینه‌ساز SGD



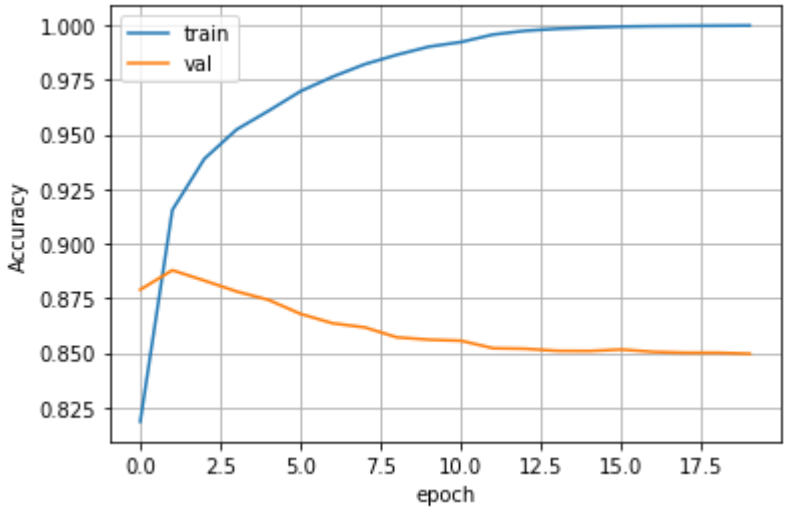



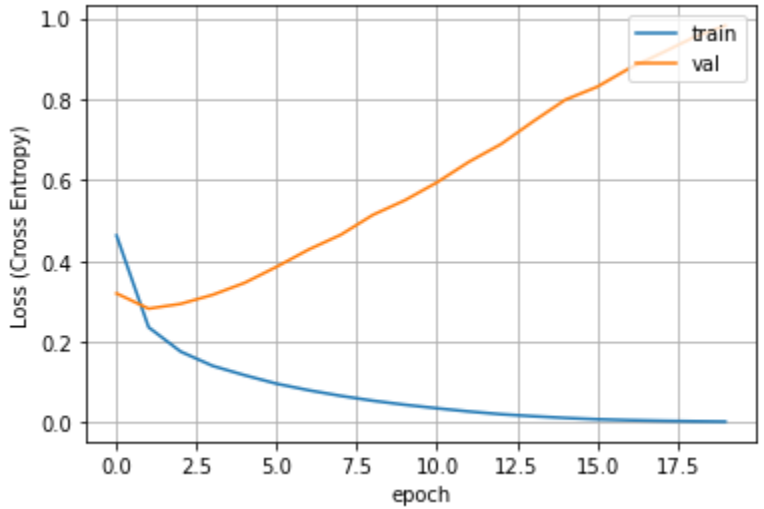



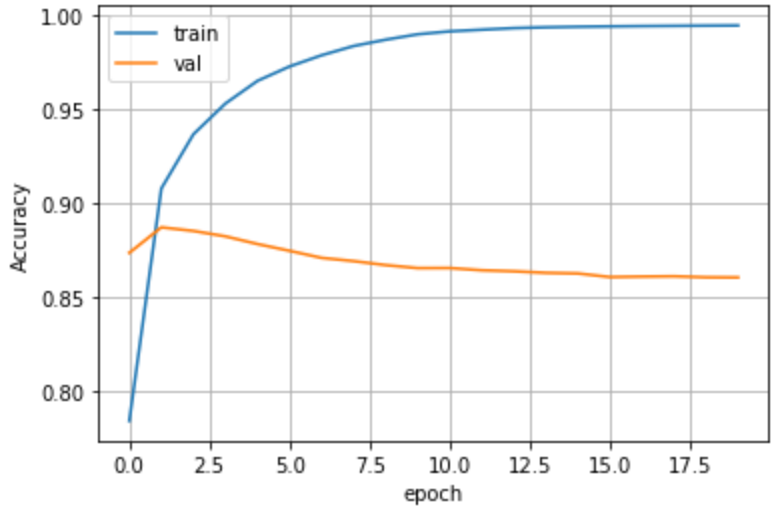



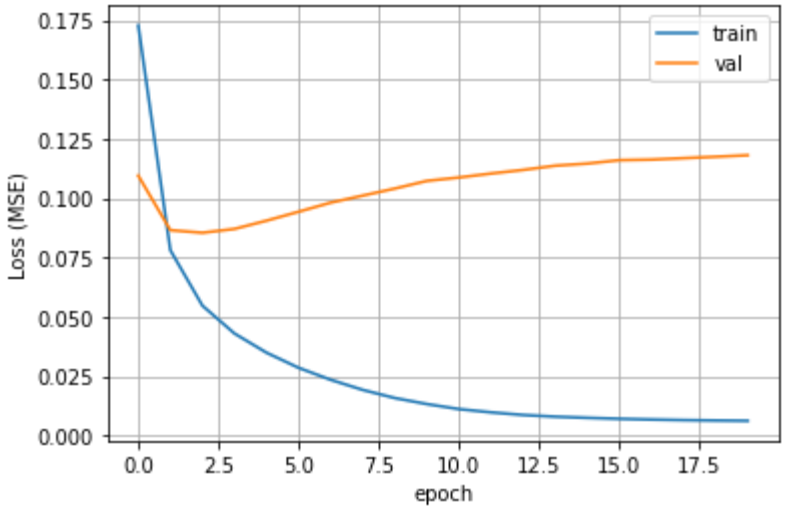
# دسته‌بندی نظرها

- بهینه‌ساز Adam



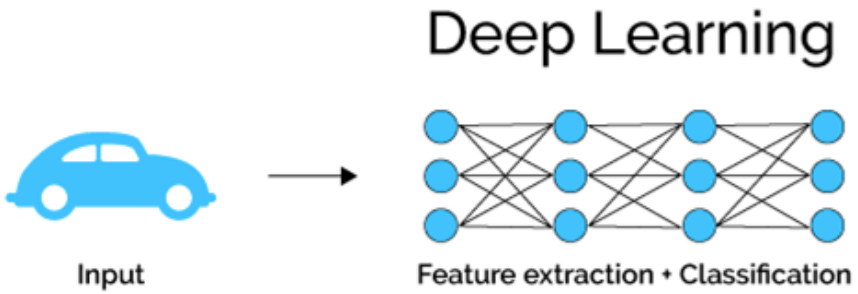
# دسته‌بندی چندکلاسه

- در لایه انتهای شبکه احتمال پسین مربوط به هر کلاس را تخمین می‌زنیم $P(y = i|\boldsymbol{x}) \in [0,1]$
- ابتدا با استفاده از یک لایه خطی احتمال غیرنرمالیزه را پیش‌بینی می‌کنیم $\boldsymbol{z} = \boldsymbol{W}^T \boldsymbol{h} + \boldsymbol{b}$
- لازم است مجموع احتمال پسین کلاس‌ها برابر با ۱ باشد و هر کدام نامنفی باشند
- تابع فعال‌سازی Softmax تعمیم تابع Sigmoid است

$$\hat{y}_i = \text{softmax}(\boldsymbol{z})_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

- تابع ضرر متناسب با این تابع فعال‌سازی، cross-entropy است

$$J(\boldsymbol{\theta}) = -\sum_{i=1}^{C} y_i \log \hat{y}_i$$



# دسته‌بندی اخبار

- مجموعه داده Reuters
  - یک مجموعه از اخبار کوتاه و عنوان آنها
  - دسته‌بندی اخبار به ۴۶ موضوع منحصر به فرد

# دسته‌بندی اخبار

```
from keras.datasets import reuters
(train_data, train_labels), (test_data,test_labels) = reuters.load_data(num_words=10000)
```

- ۸،۹۸۲ نمونه آموزشی و ۲،۲۴۶ نمونه آزمون

```
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)
```

```
y_train = keras.utils.to_categorical(train_labels)
y_test = keras.utils.to_categorical(test_labels)
```

# دسته‌بندی اخبار

- یک شبکه ۳ لایه

```
model = keras.models.Sequential()
model.add(keras.layers.Dense(64, activation='relu', input_shape=(10000,)))
model.add(keras.layers.Dense(64, activation='relu'))
model.add(keras.layers.Dense(46, activation='softmax'))

model.compile(optimizer='adam',
              loss='categorical_crossentropy',
              metrics=['accuracy'])

history = model.fit(x_train, y_train,
                    validation_data=(x_test, y_test),
                    epochs=20,
                    batch_size=512)
```

# دسته‌بندی اخبار