Evaluating NN with CV

young

2019 7 9

This is for how to apply CVed evaluation with simple NN.

load dataset

```
library(keras)

rt=dataset_reuters(num_words=10000)

c(c(train_x,train_y),c(test_x,test_y)) %<-% rt
```

preprocessing data

```
vec_seq=function(seq,dim=10000){
  results=matrix(0,nrow=length(seq),ncol=10000)
  for(i in 1:length(seq))
    results[i,seq[[i]]]<-1
  results
}

train_x=vec_seq(train_x)
test_x=vec_seq(test_x)

train_y=to_categorical(train_y)
test_y=to_categorical(test_y)
```

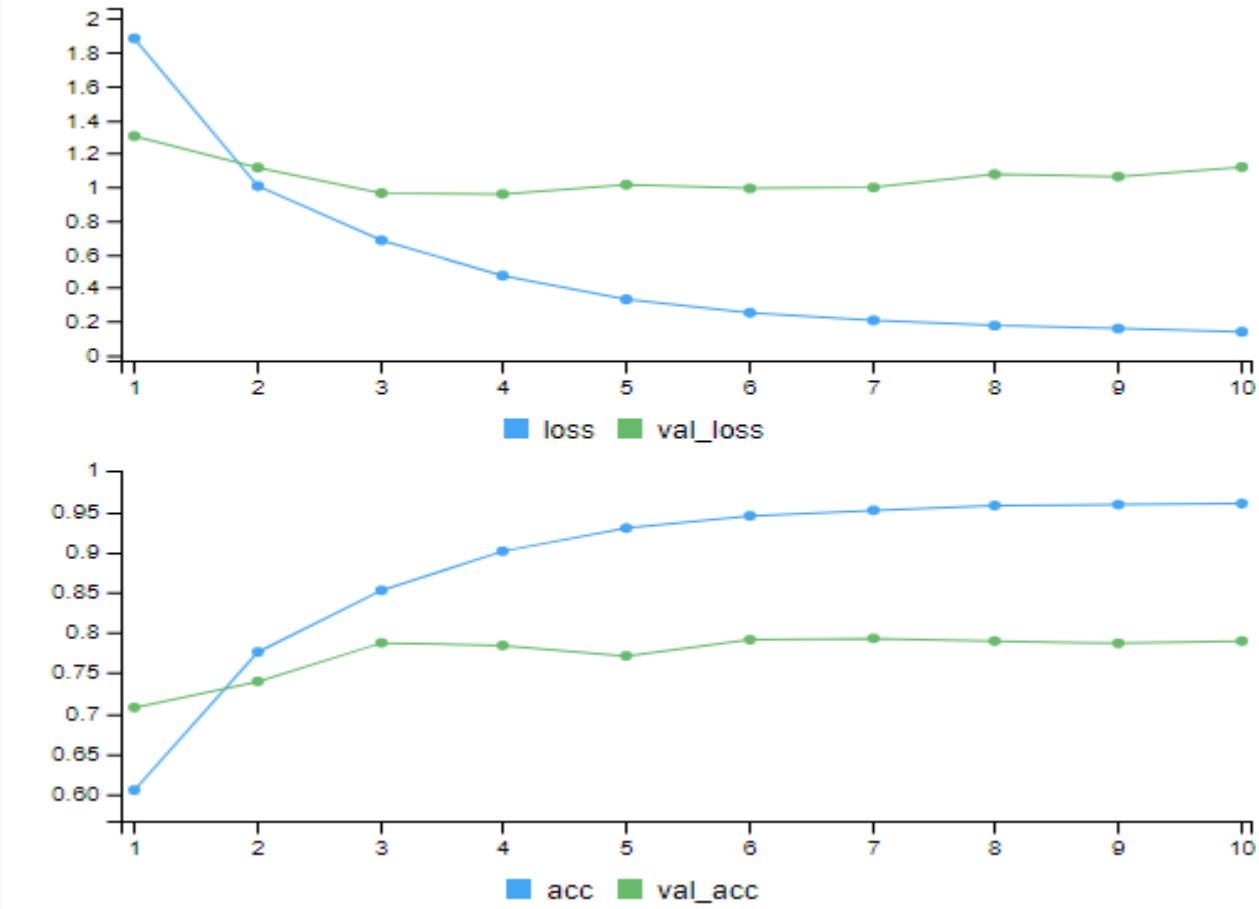
build model preset for CV

```
model_build=function(){

model=keras_model_sequential()

model %>%
  layer_dense(units=64,activation='relu',input_shape=c(10000)) %>%
  layer_dense(units=64,activation='relu') %>%
  layer_dense(units=46,activation='softmax')

model %>% compile(
  optimizer='rmsprop',
  loss='categorical_crossentropy',
```

```
  metrics=c('acc')
)
}

model = model_build()
```

You can check model this point.

```
hisory=model %>% fit(train_x,train_y,
                     batch_size=128,
                     epoch=10,
                     validation_split=0.2)
```




```
model %>% evaluate(test_x,test_y)

## $loss
## [1] 1.196157
## $acc
## [1] 0.7809439
```

prepare CV method

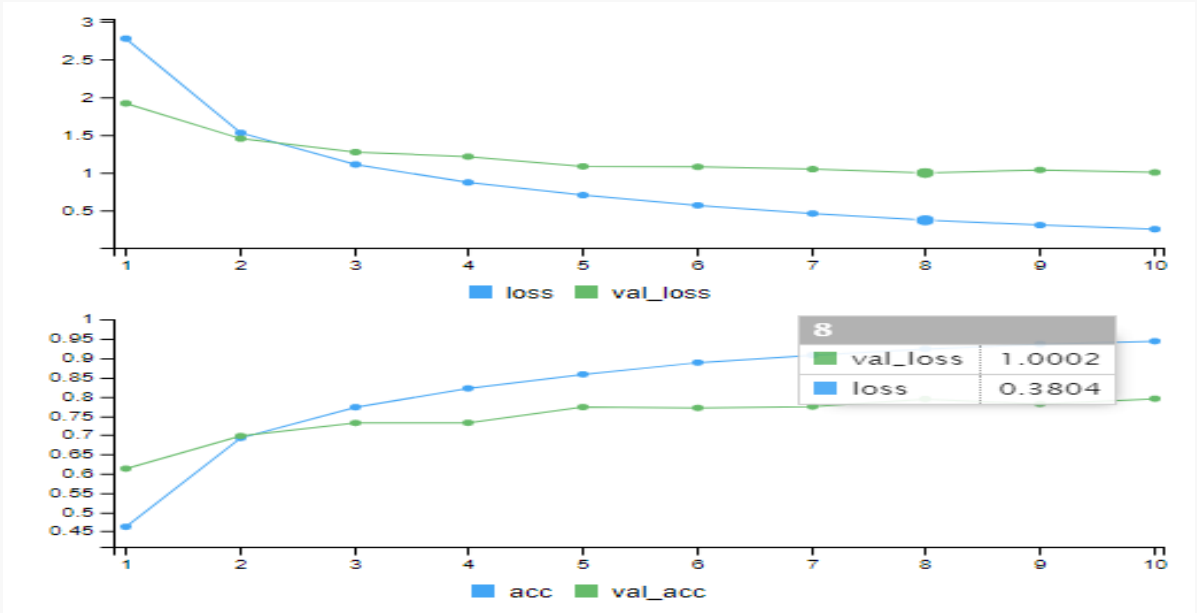
k means k-fold

```
k=4
indices=sample(1:nrow(train_x))
folds=cut(1:length(indices),breaks=k,labels = F)
results=NULL
```

build & evaluate model

```
for(i in 1:k){
val_indices=which(folds==i,arr.ind=T)

partial_train_x=train_x[-val_indices,]
partial_train_y=train_y[-val_indices,]

val_x=train_x[val_indices,]
val_y=train_y[val_indices,]

model=model_build()

history=model %>% fit(partial_train_x,partial_train_y,epoch=10,batch_size=512,
validation_data=list(val_x,val_y))

test=model %>% evaluate(test_x,test_y)
results[i]=test$acc
}
```

```
<1 of 4>
```



check CV result

```
results
```

```
## [1] 0.7760463 0.7702582 0.7871772 0.7756011
```

Compare result with previous one