

Income_Prediction_Report

April 21, 2020

1 Income prediction based on census data

1.1 Introduction

The prediction task is to determine whether a person makes over 50K a year.

In this report, I will use logistic regression and deep neural network (DNN model).

1.2 Dataset

This dataset is downloaded from Kaggle.

Data resource: UCI machine learning repository

1.3 import some libraries and our dataset.

```
[1]: # Import some libraries
import pandas as pd
import numpy as np
import matplotlib as mlt
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn
```

```
[2]: # Import datasets
train_data=pd.read_csv('/Users/Stylewsxcde991/Desktop/      /qbs-competition-1/
↳data/train.csv',index_col=0)
X_test=pd.read_csv('/Users/Stylewsxcde991/Desktop/      /qbs-competition-1/data/
↳test.csv',index_col=0)
```

1.4 Look some basic information of our dataset.

```
[3]: # Some basic information of training data
print('The shape of training data: ' + str(train_data.shape))
print('')
```

```

print('The shape of training data: ' + str(X_test.shape))
print('')
print('Basic information of our training data: ')
print(train_data.info())
print('')
print('Basic information of our testing data: ')
print(X_test.info())

```

The shape of training data: (29514, 15)

The shape of training data: (19328, 14)

Basic information of our training data:

<class 'pandas.core.frame.DataFrame'>

Int64Index: 29514 entries, 2 to 48841

Data columns (total 15 columns):

#	Column	Non-Null Count	Dtype
0	Age	29514 non-null	int64
1	Workclass	27665 non-null	object
2	fnlwgt	29514 non-null	int64
3	Education	29514 non-null	object
4	Education_Num	29514 non-null	int64
5	Martial_Status	29514 non-null	object
6	Occupation	27657 non-null	object
7	Relationship	29514 non-null	object
8	Race	29514 non-null	object
9	Sex	29514 non-null	object
10	Capital_Gain	29514 non-null	int64
11	Capital_Loss	29514 non-null	int64
12	Hours_per_week	29514 non-null	int64
13	Country	28988 non-null	object
14	Target	29514 non-null	int64

dtypes: int64(7), object(8)

memory usage: 3.6+ MB

None

Basic information of our testing data:

<class 'pandas.core.frame.DataFrame'>

Int64Index: 19328 entries, 1 to 48842

Data columns (total 14 columns):

#	Column	Non-Null Count	Dtype
0	Age	19328 non-null	int64
1	Workclass	18378 non-null	object
2	fnlwgt	19328 non-null	int64
3	Education	19328 non-null	object

```

4   Education_Num    19328 non-null   int64
5   Martial_Status   19328 non-null   object
6   Occupation       18376 non-null   object
7   Relationship     19328 non-null   object
8   Race             19328 non-null   object
9   Sex              19328 non-null   object
10  Capital_Gain     19328 non-null   int64
11  Capital_Loss     19328 non-null   int64
12  Hours_per_week   19328 non-null   int64
13  Country          18997 non-null   object
dtypes: int64(6), object(8)
memory usage: 2.2+ MB
None

```

According to above information, the shape of training data is (29514, 15) and the shape of test data is (19328, 14).

Furthermore, notice that we have missing data problem in our training dataset and test dataset (there are Null value in some features).

In particular, we have to deal with the missing data problem of 'Workclass', 'Occupation', 'Country' in our training dataset and testing dataset. We can deal with this problem by replacing all Null value with 'unknown'.

1.5 Deal with missing data

```

[4]: # Deal with missing data
train_data.Workclass=train_data.Workclass.fillna('unknown')
train_data.Occupation=train_data.Occupation.fillna('unknown')
train_data.Country=train_data.Country.fillna('unknown')
X_test.Workclass=X_test.Workclass.fillna('unknown')
X_test.Occupation=X_test.Occupation.fillna('unknown')
X_test.Country=X_test.Country.fillna('unknown')

```

1.6 Proportion of each target class (make over 50k a year or not).

```

[5]: # The proportion of each target class
NotOver50k,Over50k = train_data.Target.value_counts()
print(f'NotOver50k {NotOver50k}')
print(f'Over50k {Over50k}')
print(f'Over50k proportion {round((100*Over50k/(Over50k+NotOver50k)),2)}%')
plt.figure(figsize=(10,5))
sns.countplot(train_data['Target'])

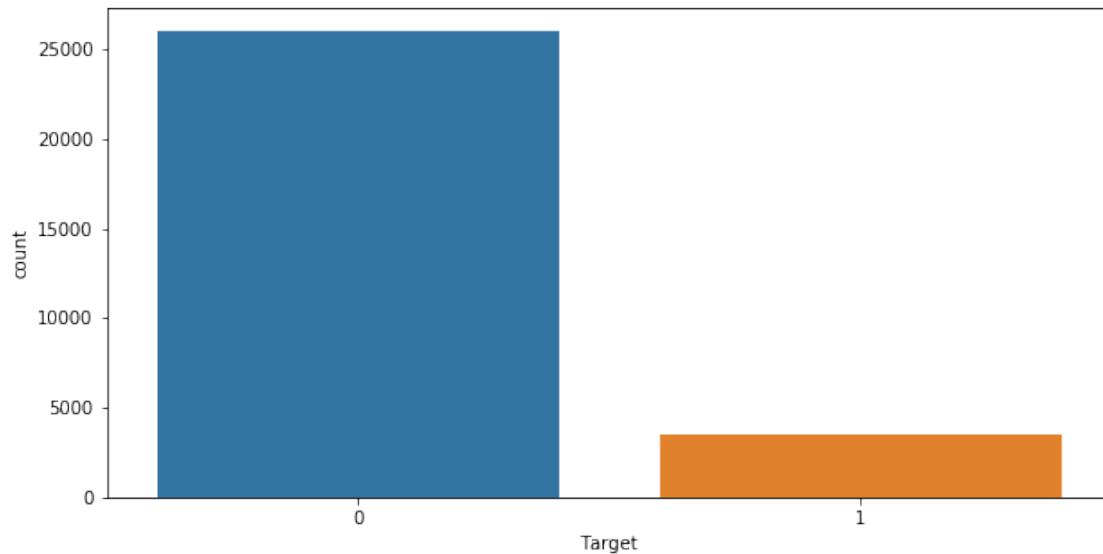
```

```

NotOver50k 26008
Over50k 3506
Over50k proportion 11.88%

```

[5]: <matplotlib.axes._subplots.AxesSubplot at 0x1088f3f10>



According to above calculation and plot, there are only 11.88% samples makes over 50K a year. Therefore, our training dataset is quiet imbalanced.

1.7 Exploratory data analysis for numerical features

As following, we analyze the correlation coefficients between our numerical features.

```
[6]: # EDA for numerical features
# data.corr()
plt.figure(figsize=(10,8))
sns.heatmap(train_data.corr(),cmap='Accent',annot=True)
plt.title('Heatmap showing correlations between numerical data')
```

[6]: Text(0.5, 1, 'Heatmap showing correlations between numerical data')



One thing to note is that the correlation coefficient between ‘fnlwgt’ and our target is quiet small (which is -0.01).

Therefore, I don’t consider ‘fnlwgt’ in my NN models.

(In fact, I have tried to incorporate ‘fnlwgt’ in my NN models and got really bad results.)

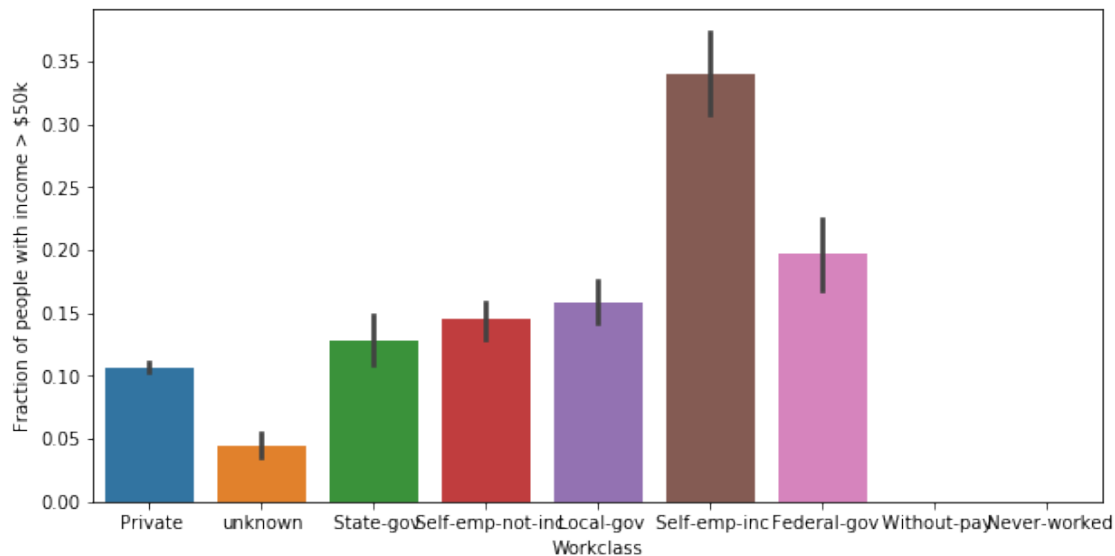
In addition, because I think ‘Education’ and ‘Education_Num’ contain the same information, I only use ‘Education_Num’ in my NN models.

1.8 Exploratory data analysis for categorical features.

```
[7]: # Drop 'fnlwgt' & 'Education'
train_data = train_data.drop(columns=['fnlwgt', 'Education'])
X_test = X_test.drop(columns=['fnlwgt', 'Education'])
```

```
[8]: # EDA for categorical features
plt.figure(figsize=(10,5))
ax = sns.barplot(x='Workclass',y='Target',data=train_data)
ax.set(ylabel='Fraction of people with income > $50k')
```

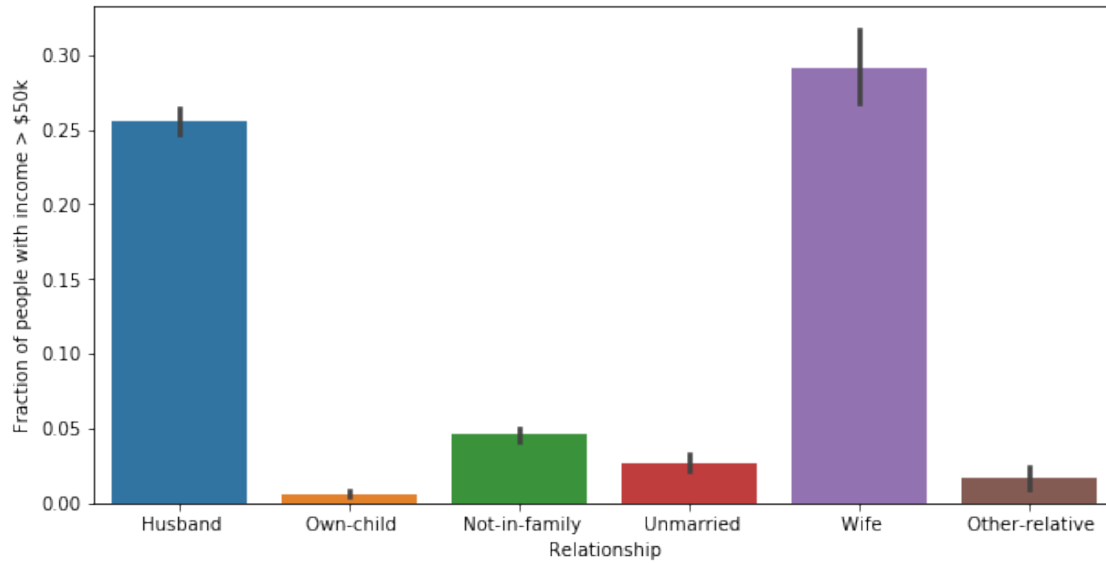
```
[8]: [Text(0, 0.5, 'Fraction of people with income > $50k')]
```



As above, people who are 'Self-emp-inc' are more likely makes over 50K a year.

```
[9]: plt.figure(figsize=(10,5))
ax = sns.barplot(x='Relationship',y='Target',data=train_data)
ax.set(ylabel='Fraction of people with income > $50k')
```

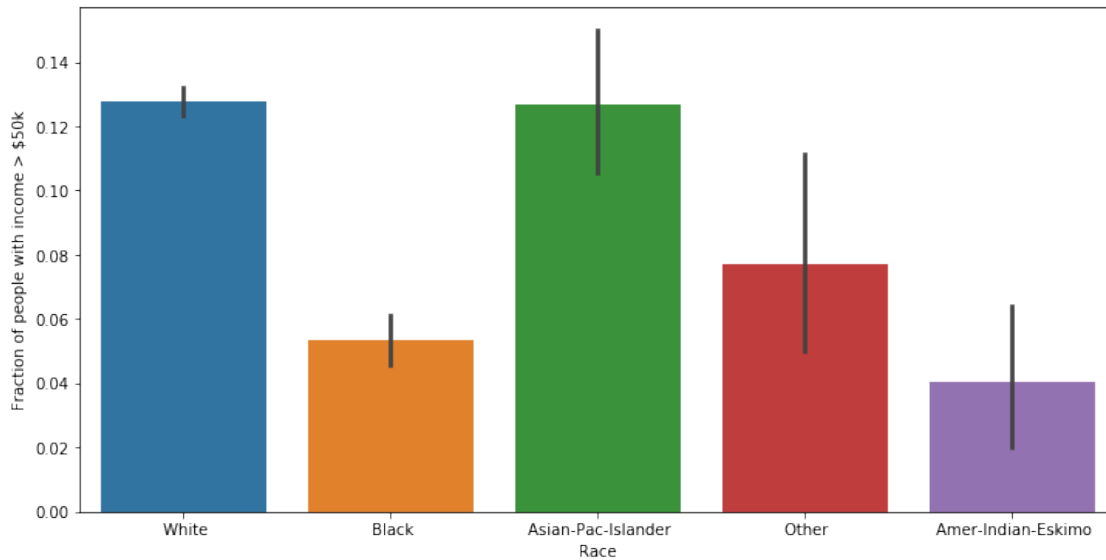
```
[9]: [Text(0, 0.5, 'Fraction of people with income > $50k')]
```



As above, 'Husband' and 'Wife' are more likely makes over 50K a year.

```
[10]: plt.figure(figsize=(12,6))
      ax=sns.barplot(x='Race',y='Target',data=train_data)
      ax.set(ylabel='Fraction of people with income > $50k')
```

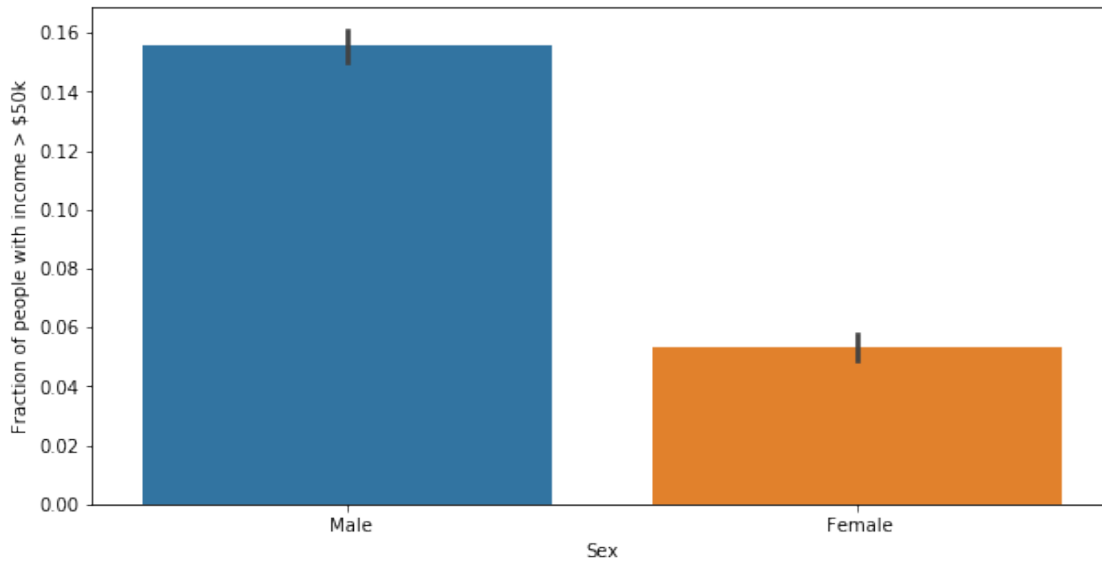
```
[10]: [Text(0, 0.5, 'Fraction of people with income > $50k')]
```



As above, 'White' and 'Asian-Pac-Islander Race' have higher proportion of people who make over 50K a year.

```
[11]: plt.figure(figsize=(10,5))
ax = sns.barplot(x='Sex',y='Target',data=train_data)
ax.set(ylabel='Fraction of people with income > $50k')
```

```
[11]: [Text(0, 0.5, 'Fraction of people with income > $50k')]
```



As above, 'Male' group has higher proportion of people who make over 50K a year.

1.9 Split the training data into features (X) and label (y).

```
[12]: # Split our train_data
X_train=train_data.iloc[:, :-1]
y_train=train_data.iloc[:, -1]
```

Now, we have to deal with the issue of categorical features.

In our training data and testing data, we have many categorical features ('Work-class', 'Marital_Status', 'Education', 'Occupation', 'Relationship', 'Race', 'Sex', 'Country'). Because our NN models can only deal with numbers, we have to encode these categorical features into numbers.

In fact, there are many different ways to encode categorical features. The method I used is so-called 'One-hot encoding' (as following).

1.10 One-hot encoding

```
[13]: # Use 'One-hot encoding' to encode categorical features.
X = X_train.append(X_test)
X = pd.get_dummies(X)
X_train = X[:29514]
X_test = X[29514:]
```

2 Standardize the data

```
[14]: from sklearn.preprocessing import StandardScaler
sc = StandardScaler().fit(X_train)
X_train=sc.transform(X_train)
X_test=sc.transform(X_test)
```

Now, we transform our datasets from dataframes to arrays, so we can feed them into NN models.

In addition, we use 7500 samples in our training dataset to be our validation set and use other samples to train our NN models.

```
[15]: # Change dataframes to arrays
X_train = np.asarray(X_train)
X_test = np.asarray(X_test)
y_train = np.asarray(y_train).astype('float32')

# validation set
X_valid = X_train[:7500]
partial_X_train = X_train[7500:]
y_valid = y_train[:7500]
partial_y_train = y_train[7500:]
```

2.1 Logistic Regression

So far, we have dealt with all issues of data pre-processing.

Now, we can start to build our logistic regression model.

```
[20]: # logistic model
from sklearn.linear_model import LogisticRegression
logistic_model = LogisticRegression(random_state=0).fit(X_train, y_train)
```

```
[21]: # Compute the confusion matrix
y_preds = logistic_model.predict(X_valid)
y_preds = (y_preds>0.5).astype(int)
print('confusion matrix')
```

```

print(sklearn.metrics.confusion_matrix(y_valid,y_preds))

# Compute accuracy
print('accuracy = '+ str(sklearn.metrics.accuracy_score(y_valid,y_preds)))

# Compute Precision
print('precision = '+str(sklearn.metrics.precision_score(y_valid,y_preds)))

# Compute Recall
print('recall = '+ str(sklearn.metrics.recall_score(y_valid,y_preds)))

# Compute F1 score
print('F1 score = '+ str(sklearn.metrics.f1_score(y_valid,y_preds)))

```

```

confusion matrix
[[6496  118]
 [ 528  358]]
accuracy = 0.9138666666666667
precision = 0.7521008403361344
recall = 0.4040632054176072
F1 score = 0.5256975036710719

```

2.2 Deep neural network (DNN model)

Now, we can start to build our NN models.

DL model draft:

3 hidden layers in this model.

The first hidden layer: 16 units with 'relu' activation function.

The second hidden layer: 16 units with 'relu' activation function.

The third hidden layer: 1 unit with 'sigmoid' activation function.

I choose the 'rmsprop' optimizer, 'binary_crossentropy' loss function, and the 'accuracy' metrics.

parameter initialization:

I use 200 epochs to train my model. The batch_size of my model is 512.

parameter tuning:

Because I think this model did well in my training dataset, I didn't tune it's parameters.

To begin with, we build our NN model with very simple structure as following.

```

[22]: # Construct our model
from keras import models
from keras import layers

```

```

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

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

```

Using TensorFlow backend.

Now we can start to fit our NN model and record all information in 'history'.

```

[23]: # Iterate on your training data by calling the fit() method of your model
history = model.fit(partial_X_train,
                    partial_y_train,
                    epochs=200,
                    batch_size=512,
                    validation_data=(X_valid, y_valid))

```

Train on 22014 samples, validate on 7500 samples

Epoch 1/200

22014/22014 [=====] - 1s 25us/step - loss: 0.5131 - accuracy: 0.8166 - val_loss: 0.3800 - val_accuracy: 0.8819

Epoch 2/200

22014/22014 [=====] - 0s 8us/step - loss: 0.3256 - accuracy: 0.8810 - val_loss: 0.2898 - val_accuracy: 0.8824

Epoch 3/200

22014/22014 [=====] - 0s 8us/step - loss: 0.2752 - accuracy: 0.8828 - val_loss: 0.2643 - val_accuracy: 0.8857

Epoch 4/200

22014/22014 [=====] - 0s 11us/step - loss: 0.2577 - accuracy: 0.8902 - val_loss: 0.2540 - val_accuracy: 0.8937

Epoch 5/200

22014/22014 [=====] - 0s 8us/step - loss: 0.2487 - accuracy: 0.8937 - val_loss: 0.2478 - val_accuracy: 0.9004

Epoch 6/200

22014/22014 [=====] - 0s 8us/step - loss: 0.2429 - accuracy: 0.8990 - val_loss: 0.2441 - val_accuracy: 0.9035

Epoch 7/200

22014/22014 [=====] - 0s 9us/step - loss: 0.2385 - accuracy: 0.9003 - val_loss: 0.2405 - val_accuracy: 0.9048

Epoch 8/200

22014/22014 [=====] - 0s 8us/step - loss: 0.2348 - accuracy: 0.9020 - val_loss: 0.2381 - val_accuracy: 0.9052

Epoch 9/200

22014/22014 [=====] - 0s 14us/step - loss: 0.2320 -

```

accuracy: 0.9027 - val_loss: 0.2362 - val_accuracy: 0.9047
Epoch 10/200
22014/22014 [=====] - 0s 9us/step - loss: 0.2294 -
accuracy: 0.9036 - val_loss: 0.2341 - val_accuracy: 0.9075
Epoch 11/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2272 -
accuracy: 0.9047 - val_loss: 0.2324 - val_accuracy: 0.9067
Epoch 12/200
22014/22014 [=====] - 0s 9us/step - loss: 0.2253 -
accuracy: 0.9056 - val_loss: 0.2310 - val_accuracy: 0.9075
Epoch 13/200
22014/22014 [=====] - 0s 10us/step - loss: 0.2236 -
accuracy: 0.9052 - val_loss: 0.2299 - val_accuracy: 0.9096
Epoch 14/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2222 -
accuracy: 0.9067 - val_loss: 0.2288 - val_accuracy: 0.9101
Epoch 15/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2210 -
accuracy: 0.9081 - val_loss: 0.2279 - val_accuracy: 0.9101
Epoch 16/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2199 -
accuracy: 0.9081 - val_loss: 0.2285 - val_accuracy: 0.9092
Epoch 17/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2189 -
accuracy: 0.9090 - val_loss: 0.2272 - val_accuracy: 0.9096
Epoch 18/200
22014/22014 [=====] - 0s 7us/step - loss: 0.2181 -
accuracy: 0.9088 - val_loss: 0.2268 - val_accuracy: 0.9103
Epoch 19/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2174 -
accuracy: 0.9096 - val_loss: 0.2269 - val_accuracy: 0.9092
Epoch 20/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2168 -
accuracy: 0.9092 - val_loss: 0.2263 - val_accuracy: 0.9092
Epoch 21/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2162 -
accuracy: 0.9098 - val_loss: 0.2256 - val_accuracy: 0.9109
Epoch 22/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2156 -
accuracy: 0.9098 - val_loss: 0.2253 - val_accuracy: 0.9113
Epoch 23/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2151 -
accuracy: 0.9111 - val_loss: 0.2256 - val_accuracy: 0.9111
Epoch 24/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2147 -
accuracy: 0.9098 - val_loss: 0.2252 - val_accuracy: 0.9111
Epoch 25/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2141 -

```

accuracy: 0.9116 - val_loss: 0.2255 - val_accuracy: 0.9107
 Epoch 26/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2136 -
 accuracy: 0.9103 - val_loss: 0.2247 - val_accuracy: 0.9120
 Epoch 27/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2132 -
 accuracy: 0.9105 - val_loss: 0.2250 - val_accuracy: 0.9123
 Epoch 28/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2132 -
 accuracy: 0.9111 - val_loss: 0.2247 - val_accuracy: 0.9115
 Epoch 29/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2125 -
 accuracy: 0.9116 - val_loss: 0.2250 - val_accuracy: 0.9115
 Epoch 30/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2121 -
 accuracy: 0.9115 - val_loss: 0.2251 - val_accuracy: 0.9112
 Epoch 31/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2118 -
 accuracy: 0.9119 - val_loss: 0.2254 - val_accuracy: 0.9104
 Epoch 32/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2114 -
 accuracy: 0.9116 - val_loss: 0.2251 - val_accuracy: 0.9113
 Epoch 33/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2110 -
 accuracy: 0.9119 - val_loss: 0.2254 - val_accuracy: 0.9096
 Epoch 34/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2107 -
 accuracy: 0.9121 - val_loss: 0.2254 - val_accuracy: 0.9107
 Epoch 35/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2104 -
 accuracy: 0.9121 - val_loss: 0.2254 - val_accuracy: 0.9112
 Epoch 36/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2100 -
 accuracy: 0.9124 - val_loss: 0.2253 - val_accuracy: 0.9107
 Epoch 37/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2097 -
 accuracy: 0.9124 - val_loss: 0.2256 - val_accuracy: 0.9120
 Epoch 38/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2092 -
 accuracy: 0.9127 - val_loss: 0.2262 - val_accuracy: 0.9111
 Epoch 39/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.2092 -
 accuracy: 0.9126 - val_loss: 0.2259 - val_accuracy: 0.9116
 Epoch 40/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.2087 -
 accuracy: 0.9133 - val_loss: 0.2263 - val_accuracy: 0.9109
 Epoch 41/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.2085 -

accuracy: 0.9136 - val_loss: 0.2260 - val_accuracy: 0.9117
Epoch 42/200
22014/22014 [=====] - 0s 13us/step - loss: 0.2081 -
accuracy: 0.9136 - val_loss: 0.2262 - val_accuracy: 0.9129
Epoch 43/200
22014/22014 [=====] - 0s 13us/step - loss: 0.2079 -
accuracy: 0.9126 - val_loss: 0.2267 - val_accuracy: 0.9115
Epoch 44/200
22014/22014 [=====] - 0s 13us/step - loss: 0.2076 -
accuracy: 0.9135 - val_loss: 0.2266 - val_accuracy: 0.9108
Epoch 45/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2071 -
accuracy: 0.9136 - val_loss: 0.2273 - val_accuracy: 0.9097
Epoch 46/200
22014/22014 [=====] - 0s 11us/step - loss: 0.2070 -
accuracy: 0.9134 - val_loss: 0.2277 - val_accuracy: 0.9105
Epoch 47/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2068 -
accuracy: 0.9134 - val_loss: 0.2271 - val_accuracy: 0.9116
Epoch 48/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2065 -
accuracy: 0.9138 - val_loss: 0.2273 - val_accuracy: 0.9104
Epoch 49/200
22014/22014 [=====] - 0s 17us/step - loss: 0.2064 -
accuracy: 0.9132 - val_loss: 0.2279 - val_accuracy: 0.9104
Epoch 50/200
22014/22014 [=====] - 0s 11us/step - loss: 0.2061 -
accuracy: 0.9133 - val_loss: 0.2280 - val_accuracy: 0.9113
Epoch 51/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2058 -
accuracy: 0.9137 - val_loss: 0.2276 - val_accuracy: 0.9109
Epoch 52/200
22014/22014 [=====] - 0s 13us/step - loss: 0.2056 -
accuracy: 0.9138 - val_loss: 0.2283 - val_accuracy: 0.9112
Epoch 53/200
22014/22014 [=====] - 0s 13us/step - loss: 0.2052 -
accuracy: 0.9137 - val_loss: 0.2293 - val_accuracy: 0.9109
Epoch 54/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2052 -
accuracy: 0.9142 - val_loss: 0.2290 - val_accuracy: 0.9107
Epoch 55/200
22014/22014 [=====] - 0s 17us/step - loss: 0.2051 -
accuracy: 0.9140 - val_loss: 0.2289 - val_accuracy: 0.9113
Epoch 56/200
22014/22014 [=====] - 0s 12us/step - loss: 0.2047 -
accuracy: 0.9141 - val_loss: 0.2288 - val_accuracy: 0.9109
Epoch 57/200
22014/22014 [=====] - 0s 17us/step - loss: 0.2044 -

accuracy: 0.9135 - val_loss: 0.2291 - val_accuracy: 0.9112
 Epoch 58/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.2043 -
 accuracy: 0.9142 - val_loss: 0.2293 - val_accuracy: 0.9105
 Epoch 59/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.2040 -
 accuracy: 0.9141 - val_loss: 0.2292 - val_accuracy: 0.9107
 Epoch 60/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.2040 -
 accuracy: 0.9140 - val_loss: 0.2289 - val_accuracy: 0.9115
 Epoch 61/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.2038 -
 accuracy: 0.9145 - val_loss: 0.2290 - val_accuracy: 0.9109
 Epoch 62/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.2035 -
 accuracy: 0.9141 - val_loss: 0.2301 - val_accuracy: 0.9092
 Epoch 63/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.2035 -
 accuracy: 0.9136 - val_loss: 0.2298 - val_accuracy: 0.9113
 Epoch 64/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.2032 -
 accuracy: 0.9148 - val_loss: 0.2303 - val_accuracy: 0.9108
 Epoch 65/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.2029 -
 accuracy: 0.9141 - val_loss: 0.2298 - val_accuracy: 0.9096
 Epoch 66/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.2028 -
 accuracy: 0.9146 - val_loss: 0.2303 - val_accuracy: 0.9103
 Epoch 67/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.2028 -
 accuracy: 0.9138 - val_loss: 0.2298 - val_accuracy: 0.9117
 Epoch 68/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.2026 -
 accuracy: 0.9137 - val_loss: 0.2303 - val_accuracy: 0.9111
 Epoch 69/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2023 -
 accuracy: 0.9143 - val_loss: 0.2303 - val_accuracy: 0.9109
 Epoch 70/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2022 -
 accuracy: 0.9140 - val_loss: 0.2307 - val_accuracy: 0.9103
 Epoch 71/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2021 -
 accuracy: 0.9142 - val_loss: 0.2303 - val_accuracy: 0.9109
 Epoch 72/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.2021 -
 accuracy: 0.9150 - val_loss: 0.2306 - val_accuracy: 0.9107
 Epoch 73/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.2016 -

accuracy: 0.9151 - val_loss: 0.2310 - val_accuracy: 0.9107
Epoch 74/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2017 -
accuracy: 0.9149 - val_loss: 0.2309 - val_accuracy: 0.9121
Epoch 75/200
22014/22014 [=====] - 0s 7us/step - loss: 0.2015 -
accuracy: 0.9147 - val_loss: 0.2313 - val_accuracy: 0.9108
Epoch 76/200
22014/22014 [=====] - 0s 7us/step - loss: 0.2015 -
accuracy: 0.9149 - val_loss: 0.2319 - val_accuracy: 0.9108
Epoch 77/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2012 -
accuracy: 0.9149 - val_loss: 0.2319 - val_accuracy: 0.9112
Epoch 78/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2012 -
accuracy: 0.9148 - val_loss: 0.2322 - val_accuracy: 0.9101
Epoch 79/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2011 -
accuracy: 0.9146 - val_loss: 0.2326 - val_accuracy: 0.9092
Epoch 80/200
22014/22014 [=====] - 0s 9us/step - loss: 0.2007 -
accuracy: 0.9158 - val_loss: 0.2322 - val_accuracy: 0.9107
Epoch 81/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2008 -
accuracy: 0.9146 - val_loss: 0.2327 - val_accuracy: 0.9099
Epoch 82/200
22014/22014 [=====] - 0s 9us/step - loss: 0.2005 -
accuracy: 0.9146 - val_loss: 0.2335 - val_accuracy: 0.9107
Epoch 83/200
22014/22014 [=====] - 0s 7us/step - loss: 0.2006 -
accuracy: 0.9151 - val_loss: 0.2330 - val_accuracy: 0.9097
Epoch 84/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2002 -
accuracy: 0.9152 - val_loss: 0.2336 - val_accuracy: 0.9116
Epoch 85/200
22014/22014 [=====] - 0s 8us/step - loss: 0.2001 -
accuracy: 0.9153 - val_loss: 0.2337 - val_accuracy: 0.9095
Epoch 86/200
22014/22014 [=====] - 0s 9us/step - loss: 0.2001 -
accuracy: 0.9154 - val_loss: 0.2347 - val_accuracy: 0.9084
Epoch 87/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1999 -
accuracy: 0.9151 - val_loss: 0.2337 - val_accuracy: 0.9101
Epoch 88/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1999 -
accuracy: 0.9160 - val_loss: 0.2349 - val_accuracy: 0.9081
Epoch 89/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1997 -

accuracy: 0.9151 - val_loss: 0.2341 - val_accuracy: 0.9099
 Epoch 90/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1996 -
 accuracy: 0.9146 - val_loss: 0.2341 - val_accuracy: 0.9101
 Epoch 91/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1994 -
 accuracy: 0.9156 - val_loss: 0.2341 - val_accuracy: 0.9103
 Epoch 92/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1995 -
 accuracy: 0.9157 - val_loss: 0.2342 - val_accuracy: 0.9104
 Epoch 93/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1991 -
 accuracy: 0.9159 - val_loss: 0.2344 - val_accuracy: 0.9103
 Epoch 94/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1990 -
 accuracy: 0.9151 - val_loss: 0.2346 - val_accuracy: 0.9091
 Epoch 95/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1991 -
 accuracy: 0.9151 - val_loss: 0.2347 - val_accuracy: 0.9104
 Epoch 96/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1989 -
 accuracy: 0.9157 - val_loss: 0.2348 - val_accuracy: 0.9085
 Epoch 97/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1986 -
 accuracy: 0.9163 - val_loss: 0.2363 - val_accuracy: 0.9073
 Epoch 98/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1986 -
 accuracy: 0.9161 - val_loss: 0.2357 - val_accuracy: 0.9087
 Epoch 99/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1983 -
 accuracy: 0.9156 - val_loss: 0.2356 - val_accuracy: 0.9079
 Epoch 100/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1983 -
 accuracy: 0.9155 - val_loss: 0.2356 - val_accuracy: 0.9081
 Epoch 101/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1983 -
 accuracy: 0.9165 - val_loss: 0.2352 - val_accuracy: 0.9099
 Epoch 102/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1982 -
 accuracy: 0.9161 - val_loss: 0.2357 - val_accuracy: 0.9095
 Epoch 103/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1979 -
 accuracy: 0.9167 - val_loss: 0.2362 - val_accuracy: 0.9083
 Epoch 104/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1979 -
 accuracy: 0.9161 - val_loss: 0.2356 - val_accuracy: 0.9091
 Epoch 105/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1977 -

accuracy: 0.9163 - val_loss: 0.2360 - val_accuracy: 0.9076
 Epoch 106/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1978 -
 accuracy: 0.9164 - val_loss: 0.2357 - val_accuracy: 0.9097
 Epoch 107/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1976 -
 accuracy: 0.9166 - val_loss: 0.2358 - val_accuracy: 0.9092
 Epoch 108/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1973 -
 accuracy: 0.9166 - val_loss: 0.2362 - val_accuracy: 0.9069
 Epoch 109/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1974 -
 accuracy: 0.9161 - val_loss: 0.2360 - val_accuracy: 0.9093
 Epoch 110/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1973 -
 accuracy: 0.9171 - val_loss: 0.2360 - val_accuracy: 0.9085
 Epoch 111/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1971 -
 accuracy: 0.9171 - val_loss: 0.2370 - val_accuracy: 0.9084
 Epoch 112/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1971 -
 accuracy: 0.9168 - val_loss: 0.2369 - val_accuracy: 0.9088
 Epoch 113/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1969 -
 accuracy: 0.9169 - val_loss: 0.2370 - val_accuracy: 0.9080
 Epoch 114/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1969 -
 accuracy: 0.9168 - val_loss: 0.2368 - val_accuracy: 0.9087
 Epoch 115/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1968 -
 accuracy: 0.9158 - val_loss: 0.2373 - val_accuracy: 0.9079
 Epoch 116/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1968 -
 accuracy: 0.9169 - val_loss: 0.2366 - val_accuracy: 0.9093
 Epoch 117/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1966 -
 accuracy: 0.9166 - val_loss: 0.2379 - val_accuracy: 0.9079
 Epoch 118/200
 22014/22014 [=====] - ETA: 0s - loss: 0.1974 -
 accuracy: 0.91 - 0s 7us/step - loss: 0.1965 - accuracy: 0.9168 - val_loss:
 0.2374 - val_accuracy: 0.9092
 Epoch 119/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.1963 -
 accuracy: 0.9167 - val_loss: 0.2381 - val_accuracy: 0.9083
 Epoch 120/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1963 -
 accuracy: 0.9168 - val_loss: 0.2380 - val_accuracy: 0.9081
 Epoch 121/200

22014/22014 [=====] - 0s 8us/step - loss: 0.1962 -
 accuracy: 0.9167 - val_loss: 0.2379 - val_accuracy: 0.9087
 Epoch 122/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1960 -
 accuracy: 0.9167 - val_loss: 0.2390 - val_accuracy: 0.9072
 Epoch 123/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1960 -
 accuracy: 0.9168 - val_loss: 0.2383 - val_accuracy: 0.9073
 Epoch 124/200
 22014/22014 [=====] - 0s 14us/step - loss: 0.1960 -
 accuracy: 0.9171 - val_loss: 0.2389 - val_accuracy: 0.9056
 Epoch 125/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1957 -
 accuracy: 0.9174 - val_loss: 0.2379 - val_accuracy: 0.9075
 Epoch 126/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1956 -
 accuracy: 0.9172 - val_loss: 0.2381 - val_accuracy: 0.9059
 Epoch 127/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1957 -
 accuracy: 0.9171 - val_loss: 0.2383 - val_accuracy: 0.9067
 Epoch 128/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1957 -
 accuracy: 0.9165 - val_loss: 0.2383 - val_accuracy: 0.9079
 Epoch 129/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1953 -
 accuracy: 0.9171 - val_loss: 0.2395 - val_accuracy: 0.9076
 Epoch 130/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1953 -
 accuracy: 0.9171 - val_loss: 0.2396 - val_accuracy: 0.9044
 Epoch 131/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1954 -
 accuracy: 0.9176 - val_loss: 0.2384 - val_accuracy: 0.9081
 Epoch 132/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1953 -
 accuracy: 0.9174 - val_loss: 0.2388 - val_accuracy: 0.9083
 Epoch 133/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1952 -
 accuracy: 0.9168 - val_loss: 0.2390 - val_accuracy: 0.9073
 Epoch 134/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1950 -
 accuracy: 0.9170 - val_loss: 0.2390 - val_accuracy: 0.9067
 Epoch 135/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1950 -
 accuracy: 0.9177 - val_loss: 0.2390 - val_accuracy: 0.9061
 Epoch 136/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1949 -
 accuracy: 0.9175 - val_loss: 0.2391 - val_accuracy: 0.9076
 Epoch 137/200

22014/22014 [=====] - 0s 7us/step - loss: 0.1948 -
 accuracy: 0.9170 - val_loss: 0.2394 - val_accuracy: 0.9077
 Epoch 138/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1947 -
 accuracy: 0.9169 - val_loss: 0.2396 - val_accuracy: 0.9063
 Epoch 139/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1946 -
 accuracy: 0.9171 - val_loss: 0.2396 - val_accuracy: 0.9076
 Epoch 140/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1946 -
 accuracy: 0.9179 - val_loss: 0.2394 - val_accuracy: 0.9084
 Epoch 141/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1945 -
 accuracy: 0.9173 - val_loss: 0.2395 - val_accuracy: 0.9079
 Epoch 142/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1945 -
 accuracy: 0.9177 - val_loss: 0.2400 - val_accuracy: 0.9061
 Epoch 143/200
 22014/22014 [=====] - 0s 6us/step - loss: 0.1943 -
 accuracy: 0.9179 - val_loss: 0.2395 - val_accuracy: 0.9068
 Epoch 144/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1942 -
 accuracy: 0.9171 - val_loss: 0.2397 - val_accuracy: 0.9069
 Epoch 145/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1941 -
 accuracy: 0.9179 - val_loss: 0.2408 - val_accuracy: 0.9061
 Epoch 146/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1941 -
 accuracy: 0.9182 - val_loss: 0.2404 - val_accuracy: 0.9056
 Epoch 147/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1940 -
 accuracy: 0.9180 - val_loss: 0.2407 - val_accuracy: 0.9069
 Epoch 148/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1940 -
 accuracy: 0.9182 - val_loss: 0.2414 - val_accuracy: 0.9049
 Epoch 149/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1939 -
 accuracy: 0.9186 - val_loss: 0.2409 - val_accuracy: 0.9072
 Epoch 150/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1939 -
 accuracy: 0.9181 - val_loss: 0.2402 - val_accuracy: 0.9073
 Epoch 151/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1938 -
 accuracy: 0.9180 - val_loss: 0.2412 - val_accuracy: 0.9059
 Epoch 152/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1938 -
 accuracy: 0.9182 - val_loss: 0.2410 - val_accuracy: 0.9063
 Epoch 153/200

22014/22014 [=====] - 0s 8us/step - loss: 0.1937 -
 accuracy: 0.9177 - val_loss: 0.2410 - val_accuracy: 0.9057
 Epoch 154/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1936 -
 accuracy: 0.9176 - val_loss: 0.2419 - val_accuracy: 0.9047
 Epoch 155/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1935 -
 accuracy: 0.9180 - val_loss: 0.2413 - val_accuracy: 0.9067
 Epoch 156/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1933 -
 accuracy: 0.9181 - val_loss: 0.2415 - val_accuracy: 0.9060
 Epoch 157/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1932 -
 accuracy: 0.9177 - val_loss: 0.2412 - val_accuracy: 0.9072
 Epoch 158/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1934 -
 accuracy: 0.9177 - val_loss: 0.2422 - val_accuracy: 0.9056
 Epoch 159/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1932 -
 accuracy: 0.9186 - val_loss: 0.2416 - val_accuracy: 0.9077
 Epoch 160/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1932 -
 accuracy: 0.9182 - val_loss: 0.2421 - val_accuracy: 0.9057
 Epoch 161/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1931 -
 accuracy: 0.9185 - val_loss: 0.2423 - val_accuracy: 0.9060
 Epoch 162/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1930 -
 accuracy: 0.9184 - val_loss: 0.2433 - val_accuracy: 0.9047
 Epoch 163/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1930 -
 accuracy: 0.9183 - val_loss: 0.2430 - val_accuracy: 0.9057
 Epoch 164/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1927 -
 accuracy: 0.9187 - val_loss: 0.2443 - val_accuracy: 0.9040
 Epoch 165/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1928 -
 accuracy: 0.9181 - val_loss: 0.2431 - val_accuracy: 0.9052
 Epoch 166/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1924 -
 accuracy: 0.9189 - val_loss: 0.2431 - val_accuracy: 0.9076
 Epoch 167/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1930 -
 accuracy: 0.9178 - val_loss: 0.2435 - val_accuracy: 0.9053
 Epoch 168/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1927 -
 accuracy: 0.9186 - val_loss: 0.2430 - val_accuracy: 0.9060
 Epoch 169/200

22014/22014 [=====] - 0s 9us/step - loss: 0.1925 -
 accuracy: 0.9187 - val_loss: 0.2435 - val_accuracy: 0.9057
 Epoch 170/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1927 -
 accuracy: 0.9181 - val_loss: 0.2436 - val_accuracy: 0.9045
 Epoch 171/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1922 -
 accuracy: 0.9193 - val_loss: 0.2438 - val_accuracy: 0.9069
 Epoch 172/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.1924 -
 accuracy: 0.9184 - val_loss: 0.2433 - val_accuracy: 0.9060
 Epoch 173/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1924 -
 accuracy: 0.9186 - val_loss: 0.2439 - val_accuracy: 0.9080
 Epoch 174/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.1924 -
 accuracy: 0.9180 - val_loss: 0.2451 - val_accuracy: 0.9047
 Epoch 175/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1921 -
 accuracy: 0.9187 - val_loss: 0.2451 - val_accuracy: 0.9033
 Epoch 176/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.1922 -
 accuracy: 0.9191 - val_loss: 0.2437 - val_accuracy: 0.9076
 Epoch 177/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1921 -
 accuracy: 0.9179 - val_loss: 0.2441 - val_accuracy: 0.9059
 Epoch 178/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1921 -
 accuracy: 0.9185 - val_loss: 0.2443 - val_accuracy: 0.9057
 Epoch 179/200
 22014/22014 [=====] - 0s 14us/step - loss: 0.1921 -
 accuracy: 0.9180 - val_loss: 0.2440 - val_accuracy: 0.9065
 Epoch 180/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1919 -
 accuracy: 0.9184 - val_loss: 0.2443 - val_accuracy: 0.9056
 Epoch 181/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1918 -
 accuracy: 0.9188 - val_loss: 0.2455 - val_accuracy: 0.9049
 Epoch 182/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.1919 -
 accuracy: 0.9186 - val_loss: 0.2450 - val_accuracy: 0.9065
 Epoch 183/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.1919 -
 accuracy: 0.9183 - val_loss: 0.2458 - val_accuracy: 0.9048
 Epoch 184/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.1916 -
 accuracy: 0.9183 - val_loss: 0.2458 - val_accuracy: 0.9051
 Epoch 185/200

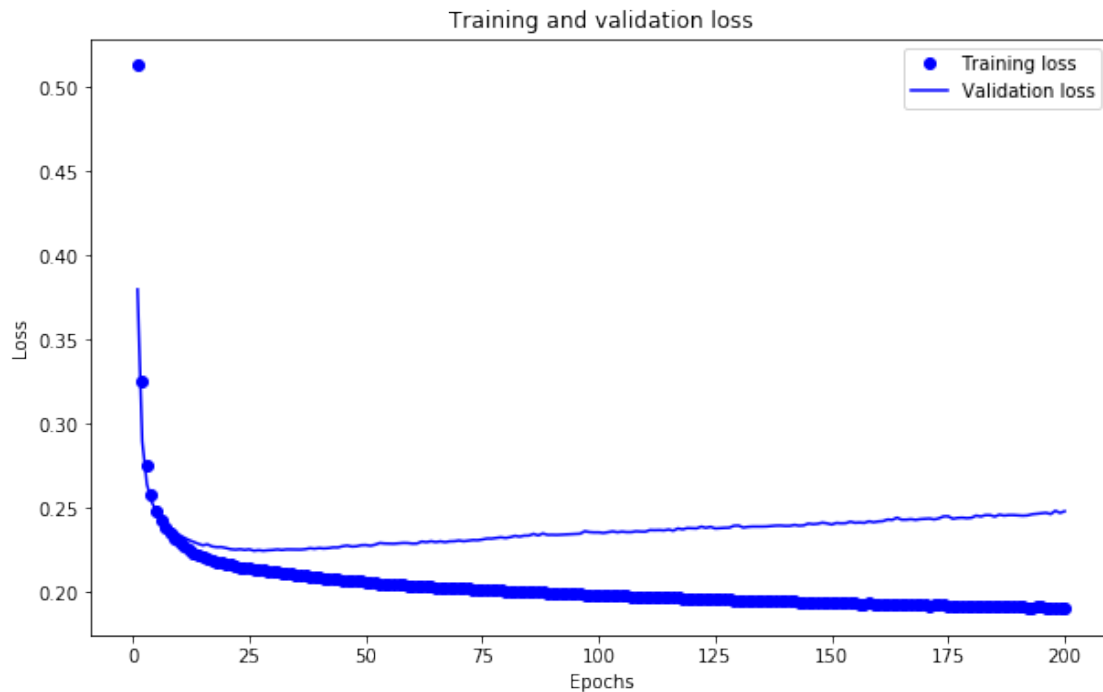
22014/22014 [=====] - 0s 12us/step - loss: 0.1915 -
accuracy: 0.9199 - val_loss: 0.2449 - val_accuracy: 0.9081
Epoch 186/200
22014/22014 [=====] - 0s 11us/step - loss: 0.1916 -
accuracy: 0.9186 - val_loss: 0.2461 - val_accuracy: 0.9040
Epoch 187/200
22014/22014 [=====] - 0s 9us/step - loss: 0.1914 -
accuracy: 0.9185 - val_loss: 0.2455 - val_accuracy: 0.9063
Epoch 188/200
22014/22014 [=====] - 0s 10us/step - loss: 0.1915 -
accuracy: 0.9186 - val_loss: 0.2459 - val_accuracy: 0.9053
Epoch 189/200
22014/22014 [=====] - 0s 10us/step - loss: 0.1915 -
accuracy: 0.9195 - val_loss: 0.2459 - val_accuracy: 0.9075
Epoch 190/200
22014/22014 [=====] - 0s 8us/step - loss: 0.1915 -
accuracy: 0.9187 - val_loss: 0.2457 - val_accuracy: 0.9055
Epoch 191/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1912 -
accuracy: 0.9188 - val_loss: 0.2455 - val_accuracy: 0.9069
Epoch 192/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1911 -
accuracy: 0.9184 - val_loss: 0.2456 - val_accuracy: 0.9064
Epoch 193/200
22014/22014 [=====] - 0s 10us/step - loss: 0.1911 -
accuracy: 0.9189 - val_loss: 0.2463 - val_accuracy: 0.9063
Epoch 194/200
22014/22014 [=====] - 0s 8us/step - loss: 0.1912 -
accuracy: 0.9191 - val_loss: 0.2466 - val_accuracy: 0.9061
Epoch 195/200
22014/22014 [=====] - 0s 8us/step - loss: 0.1913 -
accuracy: 0.9187 - val_loss: 0.2471 - val_accuracy: 0.9051
Epoch 196/200
22014/22014 [=====] - 0s 9us/step - loss: 0.1910 -
accuracy: 0.9191 - val_loss: 0.2472 - val_accuracy: 0.9044
Epoch 197/200
22014/22014 [=====] - 0s 9us/step - loss: 0.1909 -
accuracy: 0.9194 - val_loss: 0.2464 - val_accuracy: 0.9063
Epoch 198/200
22014/22014 [=====] - 0s 9us/step - loss: 0.1907 -
accuracy: 0.9186 - val_loss: 0.2484 - val_accuracy: 0.9031
Epoch 199/200
22014/22014 [=====] - 0s 10us/step - loss: 0.1909 -
accuracy: 0.9192 - val_loss: 0.2471 - val_accuracy: 0.9048
Epoch 200/200
22014/22014 [=====] - 0s 7us/step - loss: 0.1906 -
accuracy: 0.9186 - val_loss: 0.2481 - val_accuracy: 0.9036

Now, we can plot the results of loss values from the training and validation set.

```
[27]: # plot the results of loss values from the training set and validation set
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']

epochs = range(1, len(history_dict['accuracy']) + 1)

plt.figure(figsize=(10,6))
plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

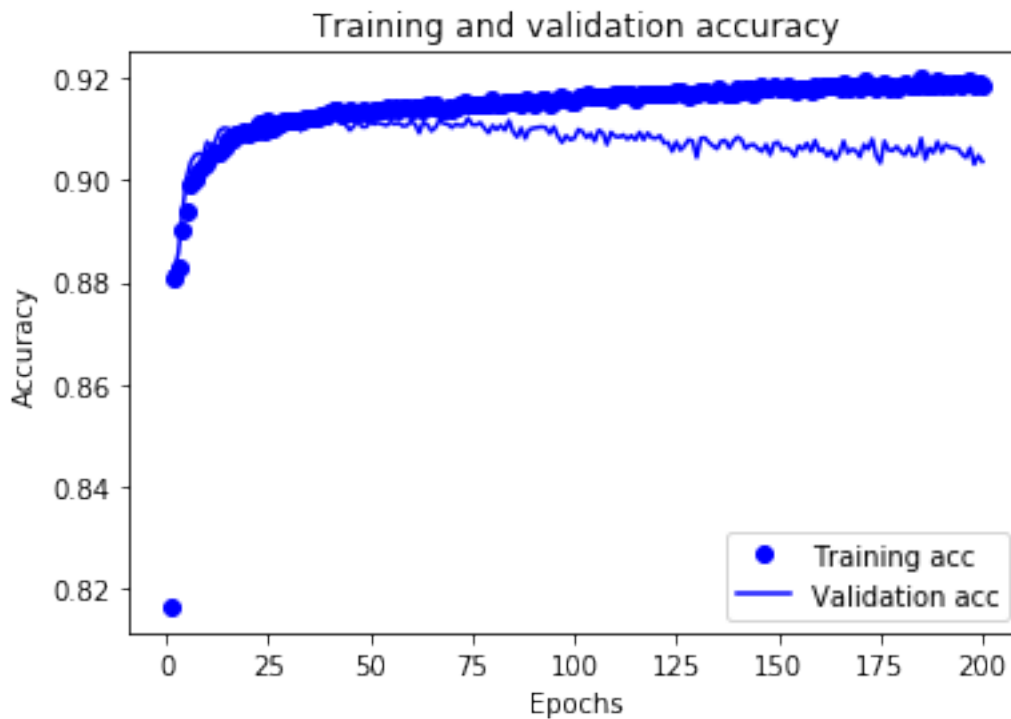


Now, we can plot the results of accuracy from the training and validation set.

```
[28]: # plot the results of accuracy from the training set and validation set
acc = history_dict['accuracy']
val_acc = history_dict['val_accuracy']
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
```



```
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



```
[29]: # Compute the confusion matrix
y_preds = model.predict(X_valid)
y_preds = (y_preds>0.5).astype(int)
print('confusion matrix')
print(sklearn.metrics.confusion_matrix(y_valid,y_preds))

# Compute accuracy
print('accuracy = '+ str(sklearn.metrics.accuracy_score(y_valid,y_preds)))

# Compute Precision
print('precision = '+str(sklearn.metrics.precision_score(y_valid,y_preds)))

# Compute Recall
print('recall = '+ str(sklearn.metrics.recall_score(y_valid,y_preds)))

# Compute F1 score
```

```
print('F1 score = '+ str(sklearn.metrics.f1_score(y_valid,y_preds)))
```

```
confusion matrix
[[6387  227]
 [ 496  390]]
accuracy = 0.9036
precision = 0.6320907617504052
recall = 0.4401805869074492
F1 score = 0.5189620758483035
```

This simple works well in our training set and validation set (with accuracy about 0.9).

However, this model didn't perform well on our test set. When I used all the training data to train this model, the public score of this model is about 0.62. Why this happened? Well, I think the reason is about our imbalanced data, there are too many 0s in 'Target'. As a result, our model prefer to give us many 0s, which leads to our fail on testing dataset.

Now, in order to deal with the issue of imbalanced dataset, I use a weighted model.

2.3 Calculate class_weights.

```
[30]: # Calculate class weight
NotOver50k, Over50k = np.bincount(train_data.Target)
total_count = len(train_data.Target)

weight_no_over50k = (1/NotOver50k)*(total_count)/2.0
weight_over50k = (1/Over50k)*(total_count)/2.0

class_weights = {0:weight_no_over50k, 1:weight_over50k}
```

Now, we can use class_weights as an argument when we construct our second NN model.

DL model draft:

3 hidden layers in this model.

The first hidden layer: 16 units with 'relu' activation function.

The second hidden layer: 16 units with 'relu' activation function.

The third hidden layer: 1 unit with 'sigmoid' activation function.

I choose the 'rmsprop' optimizer, 'binary_crossentropy' loss function, and the 'accuracy' metrics.

parameter initialization:

I use 200 epochs to train my model. The batch_size of my model is 512.

parameter tuning:

Because I think this model did well in my training dataset, I didn't tune it's parameters.

Now, let's construct our second NN model.

```
[31]: # Use weighted model!
model = models.Sequential()
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))

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

history = model.fit(partial_X_train,
                    partial_y_train,
                    epochs=200,
                    batch_size=512,
                    validation_data=(X_valid, y_valid),
                    class_weight=class_weights)
```

Train on 22014 samples, validate on 7500 samples

Epoch 1/200

22014/22014 [=====] - 0s 16us/step - loss: 0.6550 - accuracy: 0.8089 - val_loss: 0.4914 - val_accuracy: 0.7552

Epoch 2/200

22014/22014 [=====] - 0s 7us/step - loss: 0.5194 - accuracy: 0.7345 - val_loss: 0.4661 - val_accuracy: 0.7255

Epoch 3/200

22014/22014 [=====] - 0s 11us/step - loss: 0.4516 - accuracy: 0.7318 - val_loss: 0.4607 - val_accuracy: 0.7333

Epoch 4/200

22014/22014 [=====] - 0s 7us/step - loss: 0.4220 - accuracy: 0.7471 - val_loss: 0.4470 - val_accuracy: 0.7511

Epoch 5/200

22014/22014 [=====] - 0s 9us/step - loss: 0.4066 - accuracy: 0.7619 - val_loss: 0.4354 - val_accuracy: 0.7620

Epoch 6/200

22014/22014 [=====] - 0s 7us/step - loss: 0.3976 - accuracy: 0.7724 - val_loss: 0.4339 - val_accuracy: 0.7659

Epoch 7/200

22014/22014 [=====] - 0s 7us/step - loss: 0.3907 - accuracy: 0.7757 - val_loss: 0.4171 - val_accuracy: 0.7769

Epoch 8/200

22014/22014 [=====] - 0s 9us/step - loss: 0.3855 - accuracy: 0.7799 - val_loss: 0.4110 - val_accuracy: 0.7816

Epoch 9/200

22014/22014 [=====] - 0s 6us/step - loss: 0.3819 - accuracy: 0.7835 - val_loss: 0.4212 - val_accuracy: 0.7739

Epoch 10/200

22014/22014 [=====] - 0s 7us/step - loss: 0.3786 -

accuracy: 0.7831 - val_loss: 0.4191 - val_accuracy: 0.7753
 Epoch 11/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3756 -
 accuracy: 0.7864 - val_loss: 0.4191 - val_accuracy: 0.7756
 Epoch 12/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3731 -
 accuracy: 0.7875 - val_loss: 0.4078 - val_accuracy: 0.7807
 Epoch 13/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3713 -
 accuracy: 0.7871 - val_loss: 0.3982 - val_accuracy: 0.7869
 Epoch 14/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3694 -
 accuracy: 0.7893 - val_loss: 0.4198 - val_accuracy: 0.7743
 Epoch 15/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3681 -
 accuracy: 0.7882 - val_loss: 0.4114 - val_accuracy: 0.7789
 Epoch 16/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3663 -
 accuracy: 0.7906 - val_loss: 0.4115 - val_accuracy: 0.7780
 Epoch 17/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3651 -
 accuracy: 0.7897 - val_loss: 0.3990 - val_accuracy: 0.7843
 Epoch 18/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.3638 -
 accuracy: 0.7922 - val_loss: 0.4100 - val_accuracy: 0.7785
 Epoch 19/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3628 -
 accuracy: 0.7916 - val_loss: 0.4039 - val_accuracy: 0.7816
 Epoch 20/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3615 -
 accuracy: 0.7936 - val_loss: 0.4109 - val_accuracy: 0.7772
 Epoch 21/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3609 -
 accuracy: 0.7896 - val_loss: 0.4091 - val_accuracy: 0.7797
 Epoch 22/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3596 -
 accuracy: 0.7922 - val_loss: 0.4000 - val_accuracy: 0.7841
 Epoch 23/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3590 -
 accuracy: 0.7944 - val_loss: 0.4100 - val_accuracy: 0.7792
 Epoch 24/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.3576 -
 accuracy: 0.7953 - val_loss: 0.4030 - val_accuracy: 0.7833
 Epoch 25/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3576 -
 accuracy: 0.7947 - val_loss: 0.4005 - val_accuracy: 0.7831
 Epoch 26/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3563 -

accuracy: 0.7952 - val_loss: 0.3999 - val_accuracy: 0.7856
 Epoch 27/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3559 -
 accuracy: 0.7993 - val_loss: 0.4185 - val_accuracy: 0.7755
 Epoch 28/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3553 -
 accuracy: 0.7966 - val_loss: 0.4142 - val_accuracy: 0.7775
 Epoch 29/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3543 -
 accuracy: 0.7958 - val_loss: 0.4016 - val_accuracy: 0.7839
 Epoch 30/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3537 -
 accuracy: 0.7970 - val_loss: 0.3986 - val_accuracy: 0.7864
 Epoch 31/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3533 -
 accuracy: 0.7991 - val_loss: 0.4134 - val_accuracy: 0.7781
 Epoch 32/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3526 -
 accuracy: 0.7962 - val_loss: 0.3999 - val_accuracy: 0.7865
 Epoch 33/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3519 -
 accuracy: 0.7984 - val_loss: 0.3960 - val_accuracy: 0.7885
 Epoch 34/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3516 -
 accuracy: 0.7980 - val_loss: 0.3900 - val_accuracy: 0.7936
 Epoch 35/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3506 -
 accuracy: 0.8002 - val_loss: 0.4058 - val_accuracy: 0.7836
 Epoch 36/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3504 -
 accuracy: 0.8009 - val_loss: 0.4082 - val_accuracy: 0.7817
 Epoch 37/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3498 -
 accuracy: 0.7986 - val_loss: 0.3945 - val_accuracy: 0.7880
 Epoch 38/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3493 -
 accuracy: 0.7999 - val_loss: 0.3952 - val_accuracy: 0.7899
 Epoch 39/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3488 -
 accuracy: 0.8007 - val_loss: 0.3980 - val_accuracy: 0.7887
 Epoch 40/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3482 -
 accuracy: 0.7989 - val_loss: 0.3957 - val_accuracy: 0.7896
 Epoch 41/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3474 -
 accuracy: 0.8039 - val_loss: 0.4071 - val_accuracy: 0.7832
 Epoch 42/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3473 -

accuracy: 0.8014 - val_loss: 0.3987 - val_accuracy: 0.7872
 Epoch 43/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3467 -
 accuracy: 0.8018 - val_loss: 0.3944 - val_accuracy: 0.7900
 Epoch 44/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3465 -
 accuracy: 0.8012 - val_loss: 0.3842 - val_accuracy: 0.7983
 Epoch 45/200
 22014/22014 [=====] - 0s 18us/step - loss: 0.3460 -
 accuracy: 0.8046 - val_loss: 0.4052 - val_accuracy: 0.7829
 Epoch 46/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3455 -
 accuracy: 0.8042 - val_loss: 0.4120 - val_accuracy: 0.7801
 Epoch 47/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3449 -
 accuracy: 0.8023 - val_loss: 0.4007 - val_accuracy: 0.7856
 Epoch 48/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3444 -
 accuracy: 0.8019 - val_loss: 0.3967 - val_accuracy: 0.7897
 Epoch 49/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3442 -
 accuracy: 0.8017 - val_loss: 0.3938 - val_accuracy: 0.7913
 Epoch 50/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3439 -
 accuracy: 0.8049 - val_loss: 0.4079 - val_accuracy: 0.7831
 Epoch 51/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3432 -
 accuracy: 0.8039 - val_loss: 0.3904 - val_accuracy: 0.7936
 Epoch 52/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3430 -
 accuracy: 0.8048 - val_loss: 0.4065 - val_accuracy: 0.7857
 Epoch 53/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3421 -
 accuracy: 0.8026 - val_loss: 0.3872 - val_accuracy: 0.7956
 Epoch 54/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3420 -
 accuracy: 0.8063 - val_loss: 0.4032 - val_accuracy: 0.7879
 Epoch 55/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3415 -
 accuracy: 0.8054 - val_loss: 0.3986 - val_accuracy: 0.7907
 Epoch 56/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3413 -
 accuracy: 0.8057 - val_loss: 0.4005 - val_accuracy: 0.7889
 Epoch 57/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3407 -
 accuracy: 0.8062 - val_loss: 0.4029 - val_accuracy: 0.7867
 Epoch 58/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3408 -

accuracy: 0.8051 - val_loss: 0.4051 - val_accuracy: 0.7839
 Epoch 59/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3401 -
 accuracy: 0.8052 - val_loss: 0.3944 - val_accuracy: 0.7931
 Epoch 60/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3396 -
 accuracy: 0.8077 - val_loss: 0.4052 - val_accuracy: 0.7840
 Epoch 61/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3395 -
 accuracy: 0.8052 - val_loss: 0.4119 - val_accuracy: 0.7820
 Epoch 62/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3388 -
 accuracy: 0.8052 - val_loss: 0.4006 - val_accuracy: 0.7884
 Epoch 63/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3384 -
 accuracy: 0.8061 - val_loss: 0.3915 - val_accuracy: 0.7931
 Epoch 64/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3380 -
 accuracy: 0.8083 - val_loss: 0.4047 - val_accuracy: 0.7857
 Epoch 65/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3377 -
 accuracy: 0.8053 - val_loss: 0.3823 - val_accuracy: 0.8013
 Epoch 66/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3376 -
 accuracy: 0.8088 - val_loss: 0.4070 - val_accuracy: 0.7860
 Epoch 67/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3368 -
 accuracy: 0.8075 - val_loss: 0.4043 - val_accuracy: 0.7876
 Epoch 68/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3365 -
 accuracy: 0.8078 - val_loss: 0.4039 - val_accuracy: 0.7879
 Epoch 69/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3364 -
 accuracy: 0.8072 - val_loss: 0.3977 - val_accuracy: 0.7931
 Epoch 70/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3358 -
 accuracy: 0.8084 - val_loss: 0.4024 - val_accuracy: 0.7897
 Epoch 71/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3360 -
 accuracy: 0.8081 - val_loss: 0.4054 - val_accuracy: 0.7868
 Epoch 72/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3352 -
 accuracy: 0.8069 - val_loss: 0.3952 - val_accuracy: 0.7937
 Epoch 73/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3347 -
 accuracy: 0.8081 - val_loss: 0.3858 - val_accuracy: 0.7981
 Epoch 74/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3343 -

accuracy: 0.8096 - val_loss: 0.4037 - val_accuracy: 0.7861
 Epoch 75/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3343 -
 accuracy: 0.8068 - val_loss: 0.3986 - val_accuracy: 0.7915
 Epoch 76/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3335 -
 accuracy: 0.8094 - val_loss: 0.4111 - val_accuracy: 0.7845
 Epoch 77/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3334 -
 accuracy: 0.8068 - val_loss: 0.3916 - val_accuracy: 0.7952
 Epoch 78/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3326 -
 accuracy: 0.8110 - val_loss: 0.4088 - val_accuracy: 0.7863
 Epoch 79/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3326 -
 accuracy: 0.8060 - val_loss: 0.3920 - val_accuracy: 0.7969
 Epoch 80/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3324 -
 accuracy: 0.8086 - val_loss: 0.3993 - val_accuracy: 0.7935
 Epoch 81/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.3324 -
 accuracy: 0.8088 - val_loss: 0.4025 - val_accuracy: 0.7879
 Epoch 82/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3315 -
 accuracy: 0.8083 - val_loss: 0.3968 - val_accuracy: 0.7944
 Epoch 83/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3316 -
 accuracy: 0.8095 - val_loss: 0.4062 - val_accuracy: 0.7893
 Epoch 84/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3310 -
 accuracy: 0.8078 - val_loss: 0.3961 - val_accuracy: 0.7956
 Epoch 85/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3310 -
 accuracy: 0.8098 - val_loss: 0.3987 - val_accuracy: 0.7923
 Epoch 86/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3304 -
 accuracy: 0.8072 - val_loss: 0.3921 - val_accuracy: 0.7979
 Epoch 87/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3303 -
 accuracy: 0.8110 - val_loss: 0.4142 - val_accuracy: 0.7848
 Epoch 88/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3300 -
 accuracy: 0.8099 - val_loss: 0.4058 - val_accuracy: 0.7903
 Epoch 89/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.3295 -
 accuracy: 0.8091 - val_loss: 0.3999 - val_accuracy: 0.7931
 Epoch 90/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3292 -

accuracy: 0.8110 - val_loss: 0.4013 - val_accuracy: 0.7927
 Epoch 91/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3289 -
 accuracy: 0.8111 - val_loss: 0.4161 - val_accuracy: 0.7849
 Epoch 92/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3282 -
 accuracy: 0.8104 - val_loss: 0.4028 - val_accuracy: 0.7937
 Epoch 93/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3284 -
 accuracy: 0.8112 - val_loss: 0.4068 - val_accuracy: 0.7896
 Epoch 94/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3282 -
 accuracy: 0.8094 - val_loss: 0.3902 - val_accuracy: 0.7984
 Epoch 95/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3277 -
 accuracy: 0.8121 - val_loss: 0.4110 - val_accuracy: 0.7876
 Epoch 96/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3277 -
 accuracy: 0.8108 - val_loss: 0.4133 - val_accuracy: 0.7859
 Epoch 97/200
 22014/22014 [=====] - 0s 13us/step - loss: 0.3275 -
 accuracy: 0.8113 - val_loss: 0.4113 - val_accuracy: 0.7888
 Epoch 98/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3270 -
 accuracy: 0.8103 - val_loss: 0.4127 - val_accuracy: 0.7871
 Epoch 99/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3264 -
 accuracy: 0.8124 - val_loss: 0.4098 - val_accuracy: 0.7901
 Epoch 100/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3260 -
 accuracy: 0.8136 - val_loss: 0.4023 - val_accuracy: 0.7944
 Epoch 101/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3264 -
 accuracy: 0.8113 - val_loss: 0.3934 - val_accuracy: 0.8000
 Epoch 102/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3256 -
 accuracy: 0.8132 - val_loss: 0.3868 - val_accuracy: 0.8056
 Epoch 103/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3257 -
 accuracy: 0.8141 - val_loss: 0.3949 - val_accuracy: 0.7993
 Epoch 104/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3255 -
 accuracy: 0.8148 - val_loss: 0.4102 - val_accuracy: 0.7907
 Epoch 105/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3252 -
 accuracy: 0.8131 - val_loss: 0.3996 - val_accuracy: 0.7995
 Epoch 106/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3248 -

accuracy: 0.8136 - val_loss: 0.3993 - val_accuracy: 0.7995
 Epoch 107/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3246 -
 accuracy: 0.8157 - val_loss: 0.4027 - val_accuracy: 0.7956
 Epoch 108/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3242 -
 accuracy: 0.8143 - val_loss: 0.3954 - val_accuracy: 0.7996
 Epoch 109/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3240 -
 accuracy: 0.8147 - val_loss: 0.3947 - val_accuracy: 0.8024
 Epoch 110/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3240 -
 accuracy: 0.8160 - val_loss: 0.4106 - val_accuracy: 0.7893
 Epoch 111/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3232 -
 accuracy: 0.8162 - val_loss: 0.4055 - val_accuracy: 0.7932
 Epoch 112/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3233 -
 accuracy: 0.8155 - val_loss: 0.4158 - val_accuracy: 0.7885
 Epoch 113/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3229 -
 accuracy: 0.8154 - val_loss: 0.3977 - val_accuracy: 0.8007
 Epoch 114/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3230 -
 accuracy: 0.8175 - val_loss: 0.4037 - val_accuracy: 0.7948
 Epoch 115/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3225 -
 accuracy: 0.8161 - val_loss: 0.4099 - val_accuracy: 0.7907
 Epoch 116/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3221 -
 accuracy: 0.8156 - val_loss: 0.4105 - val_accuracy: 0.7921
 Epoch 117/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3218 -
 accuracy: 0.8172 - val_loss: 0.4048 - val_accuracy: 0.7935
 Epoch 118/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3216 -
 accuracy: 0.8176 - val_loss: 0.4055 - val_accuracy: 0.7940
 Epoch 119/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3212 -
 accuracy: 0.8154 - val_loss: 0.3920 - val_accuracy: 0.8040
 Epoch 120/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3211 -
 accuracy: 0.8183 - val_loss: 0.4219 - val_accuracy: 0.7848
 Epoch 121/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3211 -
 accuracy: 0.8171 - val_loss: 0.4169 - val_accuracy: 0.7884
 Epoch 122/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3205 -

accuracy: 0.8157 - val_loss: 0.3883 - val_accuracy: 0.8056
 Epoch 123/200
 22014/22014 [=====] - 0s 14us/step - loss: 0.3208 -
 accuracy: 0.8174 - val_loss: 0.3952 - val_accuracy: 0.8025
 Epoch 124/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3203 -
 accuracy: 0.8177 - val_loss: 0.4039 - val_accuracy: 0.7971
 Epoch 125/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3202 -
 accuracy: 0.8184 - val_loss: 0.4002 - val_accuracy: 0.7977
 Epoch 126/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3202 -
 accuracy: 0.8164 - val_loss: 0.4065 - val_accuracy: 0.7937
 Epoch 127/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3194 -
 accuracy: 0.8186 - val_loss: 0.4108 - val_accuracy: 0.7916
 Epoch 128/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3195 -
 accuracy: 0.8182 - val_loss: 0.4213 - val_accuracy: 0.7863
 Epoch 129/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3193 -
 accuracy: 0.8173 - val_loss: 0.4055 - val_accuracy: 0.7956
 Epoch 130/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3192 -
 accuracy: 0.8186 - val_loss: 0.4212 - val_accuracy: 0.7857
 Epoch 131/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3188 -
 accuracy: 0.8163 - val_loss: 0.3932 - val_accuracy: 0.8033
 Epoch 132/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3185 -
 accuracy: 0.8207 - val_loss: 0.4178 - val_accuracy: 0.7884
 Epoch 133/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3186 -
 accuracy: 0.8181 - val_loss: 0.4054 - val_accuracy: 0.7977
 Epoch 134/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3184 -
 accuracy: 0.8187 - val_loss: 0.4079 - val_accuracy: 0.7928
 Epoch 135/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3180 -
 accuracy: 0.8180 - val_loss: 0.4087 - val_accuracy: 0.7955
 Epoch 136/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3179 -
 accuracy: 0.8198 - val_loss: 0.4175 - val_accuracy: 0.7884
 Epoch 137/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3175 -
 accuracy: 0.8187 - val_loss: 0.4094 - val_accuracy: 0.7916
 Epoch 138/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3171 -

accuracy: 0.8192 - val_loss: 0.4013 - val_accuracy: 0.8001
 Epoch 139/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3172 -
 accuracy: 0.8190 - val_loss: 0.4034 - val_accuracy: 0.7977
 Epoch 140/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3167 -
 accuracy: 0.8180 - val_loss: 0.3886 - val_accuracy: 0.8056
 Epoch 141/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3167 -
 accuracy: 0.8199 - val_loss: 0.4022 - val_accuracy: 0.7983
 Epoch 142/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3164 -
 accuracy: 0.8194 - val_loss: 0.4010 - val_accuracy: 0.8008
 Epoch 143/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3166 -
 accuracy: 0.8209 - val_loss: 0.4115 - val_accuracy: 0.7943
 Epoch 144/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3158 -
 accuracy: 0.8199 - val_loss: 0.4164 - val_accuracy: 0.7905
 Epoch 145/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3158 -
 accuracy: 0.8169 - val_loss: 0.4018 - val_accuracy: 0.8025
 Epoch 146/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3159 -
 accuracy: 0.8201 - val_loss: 0.4056 - val_accuracy: 0.7960
 Epoch 147/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3154 -
 accuracy: 0.8211 - val_loss: 0.4081 - val_accuracy: 0.7951
 Epoch 148/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3155 -
 accuracy: 0.8191 - val_loss: 0.4019 - val_accuracy: 0.7991
 Epoch 149/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3149 -
 accuracy: 0.8203 - val_loss: 0.4086 - val_accuracy: 0.7923
 Epoch 150/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3151 -
 accuracy: 0.8211 - val_loss: 0.4127 - val_accuracy: 0.7909
 Epoch 151/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3144 -
 accuracy: 0.8202 - val_loss: 0.3916 - val_accuracy: 0.8072
 Epoch 152/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3149 -
 accuracy: 0.8222 - val_loss: 0.4100 - val_accuracy: 0.7949
 Epoch 153/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3142 -
 accuracy: 0.8214 - val_loss: 0.4152 - val_accuracy: 0.7937
 Epoch 154/200
 22014/22014 [=====] - 0s 12us/step - loss: 0.3143 -

accuracy: 0.8218 - val_loss: 0.4051 - val_accuracy: 0.7977
 Epoch 155/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3137 -
 accuracy: 0.8222 - val_loss: 0.4283 - val_accuracy: 0.7869
 Epoch 156/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3140 -
 accuracy: 0.8202 - val_loss: 0.4084 - val_accuracy: 0.7961
 Epoch 157/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3136 -
 accuracy: 0.8217 - val_loss: 0.4204 - val_accuracy: 0.7911
 Epoch 158/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3136 -
 accuracy: 0.8213 - val_loss: 0.4075 - val_accuracy: 0.7959
 Epoch 159/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3129 -
 accuracy: 0.8215 - val_loss: 0.3985 - val_accuracy: 0.8001
 Epoch 160/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3134 -
 accuracy: 0.8210 - val_loss: 0.4149 - val_accuracy: 0.7935
 Epoch 161/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3128 -
 accuracy: 0.8226 - val_loss: 0.4242 - val_accuracy: 0.7879
 Epoch 162/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3129 -
 accuracy: 0.8213 - val_loss: 0.4148 - val_accuracy: 0.7932
 Epoch 163/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3122 -
 accuracy: 0.8222 - val_loss: 0.4024 - val_accuracy: 0.7993
 Epoch 164/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3119 -
 accuracy: 0.8226 - val_loss: 0.4034 - val_accuracy: 0.7993
 Epoch 165/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3119 -
 accuracy: 0.8245 - val_loss: 0.4202 - val_accuracy: 0.7903
 Epoch 166/200
 22014/22014 [=====] - 0s 11us/step - loss: 0.3117 -
 accuracy: 0.8218 - val_loss: 0.4012 - val_accuracy: 0.8016
 Epoch 167/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3114 -
 accuracy: 0.8232 - val_loss: 0.4113 - val_accuracy: 0.7980
 Epoch 168/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3116 -
 accuracy: 0.8229 - val_loss: 0.4187 - val_accuracy: 0.7916
 Epoch 169/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3112 -
 accuracy: 0.8234 - val_loss: 0.4053 - val_accuracy: 0.7985
 Epoch 170/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3109 -

accuracy: 0.8240 - val_loss: 0.4260 - val_accuracy: 0.7877
 Epoch 171/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3109 -
 accuracy: 0.8221 - val_loss: 0.4055 - val_accuracy: 0.7969
 Epoch 172/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3106 -
 accuracy: 0.8235 - val_loss: 0.4119 - val_accuracy: 0.7947
 Epoch 173/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3106 -
 accuracy: 0.8239 - val_loss: 0.4048 - val_accuracy: 0.7983
 Epoch 174/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3103 -
 accuracy: 0.8234 - val_loss: 0.3993 - val_accuracy: 0.8028
 Epoch 175/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3106 -
 accuracy: 0.8242 - val_loss: 0.4204 - val_accuracy: 0.7933
 Epoch 176/200
 22014/22014 [=====] - 0s 9us/step - loss: 0.3101 -
 accuracy: 0.8234 - val_loss: 0.4214 - val_accuracy: 0.7941
 Epoch 177/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3098 -
 accuracy: 0.8248 - val_loss: 0.4227 - val_accuracy: 0.7936
 Epoch 178/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3099 -
 accuracy: 0.8237 - val_loss: 0.4124 - val_accuracy: 0.7968
 Epoch 179/200
 22014/22014 [=====] - 0s 7us/step - loss: 0.3096 -
 accuracy: 0.8239 - val_loss: 0.4151 - val_accuracy: 0.7941
 Epoch 180/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3093 -
 accuracy: 0.8226 - val_loss: 0.3931 - val_accuracy: 0.8084
 Epoch 181/200
 22014/22014 [=====] - 0s 10us/step - loss: 0.3091 -
 accuracy: 0.8246 - val_loss: 0.4111 - val_accuracy: 0.7981
 Epoch 182/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3091 -
 accuracy: 0.8248 - val_loss: 0.4359 - val_accuracy: 0.7848
 Epoch 183/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3092 -
 accuracy: 0.8227 - val_loss: 0.4036 - val_accuracy: 0.8012
 Epoch 184/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3088 -
 accuracy: 0.8244 - val_loss: 0.4213 - val_accuracy: 0.7939
 Epoch 185/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3087 -
 accuracy: 0.8238 - val_loss: 0.4176 - val_accuracy: 0.7927
 Epoch 186/200
 22014/22014 [=====] - 0s 8us/step - loss: 0.3090 -

```

accuracy: 0.8242 - val_loss: 0.4126 - val_accuracy: 0.7952
Epoch 187/200
22014/22014 [=====] - 0s 7us/step - loss: 0.3085 -
accuracy: 0.8244 - val_loss: 0.4098 - val_accuracy: 0.8005
Epoch 188/200
22014/22014 [=====] - 0s 12us/step - loss: 0.3087 -
accuracy: 0.8254 - val_loss: 0.4123 - val_accuracy: 0.7983
Epoch 189/200
22014/22014 [=====] - 0s 7us/step - loss: 0.3083 -
accuracy: 0.8241 - val_loss: 0.4071 - val_accuracy: 0.8003
Epoch 190/200
22014/22014 [=====] - 0s 7us/step - loss: 0.3081 -
accuracy: 0.8254 - val_loss: 0.4062 - val_accuracy: 0.8011
Epoch 191/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3083 -
accuracy: 0.8248 - val_loss: 0.4105 - val_accuracy: 0.7993
Epoch 192/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3077 -
accuracy: 0.8253 - val_loss: 0.4156 - val_accuracy: 0.7971
Epoch 193/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3075 -
accuracy: 0.8247 - val_loss: 0.4192 - val_accuracy: 0.7967
Epoch 194/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3074 -
accuracy: 0.8243 - val_loss: 0.4072 - val_accuracy: 0.8013
Epoch 195/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3079 -
accuracy: 0.8257 - val_loss: 0.4186 - val_accuracy: 0.7956
Epoch 196/200
22014/22014 [=====] - 0s 10us/step - loss: 0.3072 -
accuracy: 0.8255 - val_loss: 0.4150 - val_accuracy: 0.7976
Epoch 197/200
22014/22014 [=====] - 0s 10us/step - loss: 0.3070 -
accuracy: 0.8256 - val_loss: 0.4002 - val_accuracy: 0.8051
Epoch 198/200
22014/22014 [=====] - 0s 9us/step - loss: 0.3068 -
accuracy: 0.8246 - val_loss: 0.4035 - val_accuracy: 0.8065
Epoch 199/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3075 -
accuracy: 0.8261 - val_loss: 0.4159 - val_accuracy: 0.7979
Epoch 200/200
22014/22014 [=====] - 0s 8us/step - loss: 0.3070 -
accuracy: 0.8244 - val_loss: 0.4174 - val_accuracy: 0.7977

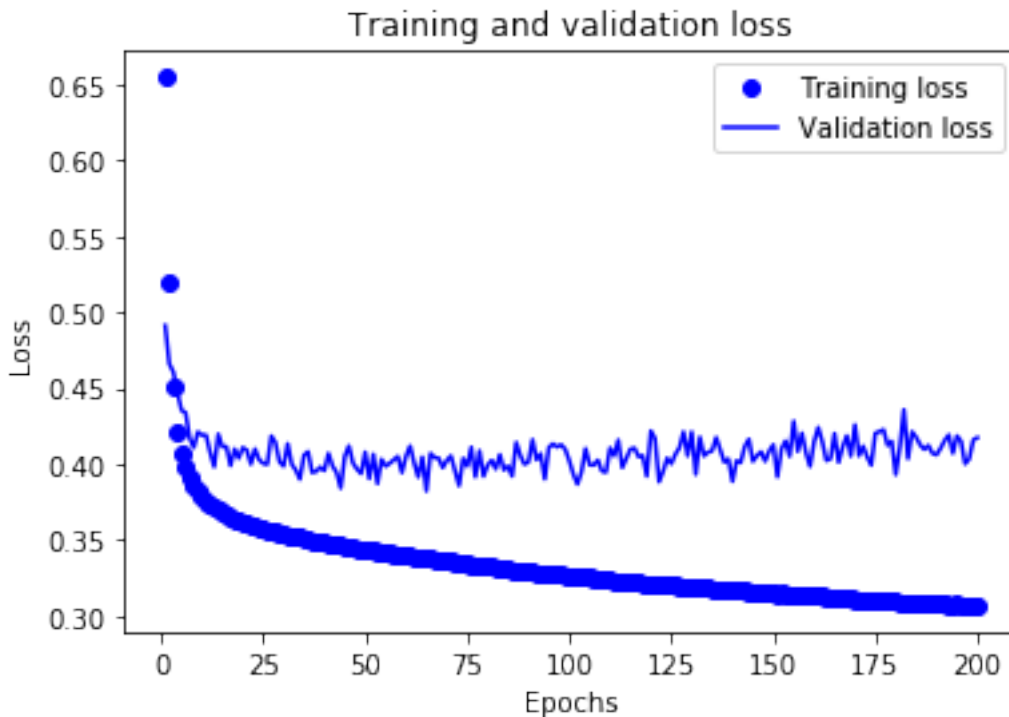
```

After constructing our weighted model, we can plot the results of loss values from the training and validation set.

```
[32]: # plot the results of loss values from the training set and validation set
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']

epochs = range(1, len(history_dict['accuracy']) + 1)

plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()
```

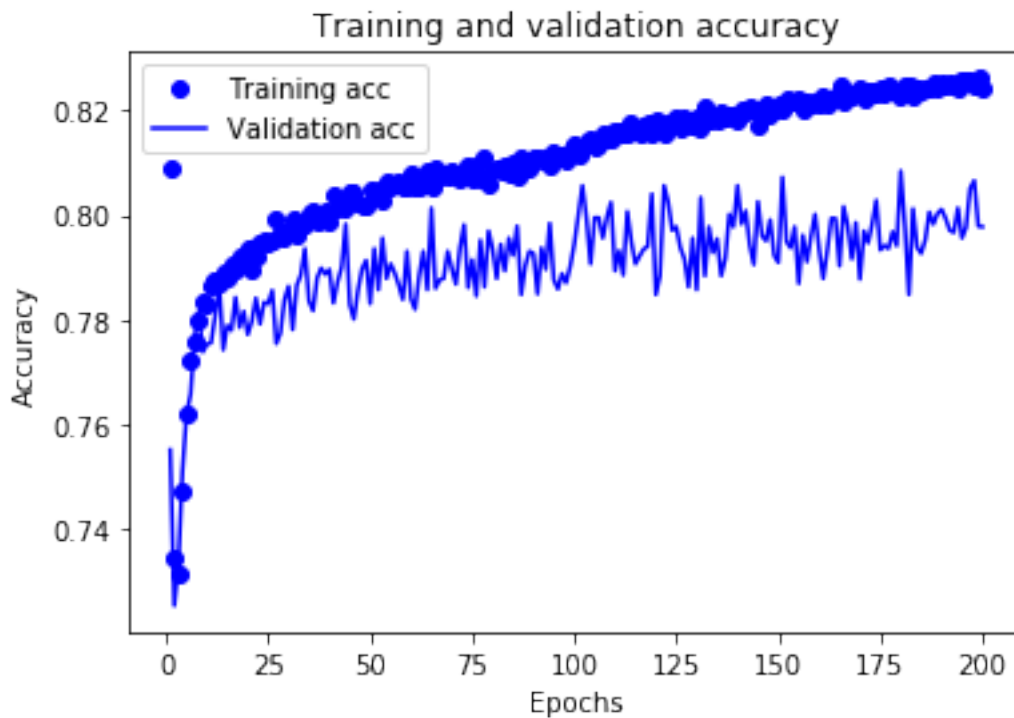


Also, we can plot the results of accuracy from the training and validation set.

```
[33]: #plt.clf()
acc = history_dict['accuracy']
val_acc = history_dict['val_accuracy']
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
```



```
plt.ylabel('Accuracy')
plt.legend()
plt.show()
```



```
[34]: # Compute the confusion matrix
y_preds = model.predict(X_valid)
y_preds = (y_preds>0.5).astype(int)
print('confusion matrix')
print(sklearn.metrics.confusion_matrix(y_valid,y_preds))

# Compute accuracy
print('accuracy = '+ str(sklearn.metrics.accuracy_score(y_valid,y_preds)))

# Compute Precision
print('precision = '+str(sklearn.metrics.precision_score(y_valid,y_preds)))

# Compute Recall
print('recall = '+ str(sklearn.metrics.recall_score(y_valid,y_preds)))

# Compute F1 score
print('F1 score = '+ str(sklearn.metrics.f1_score(y_valid,y_preds)))
```

confusion matrix

```
[[5254 1360]
 [ 157  729]]
accuracy = 0.7977333333333333
precision = 0.3489707994255625
recall = 0.8227990970654627
F1 score = 0.4900840336134454
```

When I used all the training data to train this model, the public score of this model is about 0.83 (the private score is also about 0.82). By using weighted model, we successfully overcome the problem of imbalanced training dataset.

Now, we should try to improve our performance in training data.

Let's add two layers and units.

(After we get great performance in training data, we should deal with the overfitting problem.)

```
[41]: model = models.Sequential()
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))

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

history = model.fit(partial_X_train,
                    partial_y_train,
                    epochs=200,
                    batch_size=512,
                    validation_data=(X_valid, y_valid),
                    class_weight=class_weights)

# plot the results of loss values from the training set and validation set
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']

epochs = range(1, len(history_dict['accuracy']) + 1)

plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
```

```

plt.legend()
plt.show()

#plt.clf()
acc = history_dict['accuracy']
val_acc = history_dict['val_accuracy']
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# Compute the confusion matrix
y_preds = model.predict(X_valid)
y_preds = (y_preds>0.5).astype(int)
print('confusion matrix')
print(sklearn.metrics.confusion_matrix(y_valid,y_preds))

# Compute accuracy
print('accuracy = '+ str(sklearn.metrics.accuracy_score(y_valid,y_preds)))

# Compute Precision
print('precision = '+str(sklearn.metrics.precision_score(y_valid,y_preds)))

# Compute Recall
print('recall = '+ str(sklearn.metrics.recall_score(y_valid,y_preds)))

# Compute F1 score
print('F1 score = '+ str(sklearn.metrics.f1_score(y_valid,y_preds)))

```

Train on 22014 samples, validate on 7500 samples

Epoch 1/200

22014/22014 [=====] - 1s 31us/step - loss: 0.4784 -
accuracy: 0.7330 - val_loss: 0.4848 - val_accuracy: 0.7391

Epoch 2/200

22014/22014 [=====] - 1s 26us/step - loss: 0.4038 -
accuracy: 0.7715 - val_loss: 0.3936 - val_accuracy: 0.7897

Epoch 3/200

22014/22014 [=====] - 0s 19us/step - loss: 0.3876 -
accuracy: 0.7810 - val_loss: 0.3727 - val_accuracy: 0.7991

Epoch 4/200

22014/22014 [=====] - 1s 23us/step - loss: 0.3715 -
accuracy: 0.7885 - val_loss: 0.3757 - val_accuracy: 0.7913

Epoch 5/200

22014/22014 [=====] - 0s 18us/step - loss: 0.3641 - accuracy: 0.7884 - val_loss: 0.4322 - val_accuracy: 0.7645
Epoch 6/200
22014/22014 [=====] - 0s 22us/step - loss: 0.3581 - accuracy: 0.7913 - val_loss: 0.3333 - val_accuracy: 0.8220
Epoch 7/200
22014/22014 [=====] - 0s 19us/step - loss: 0.3519 - accuracy: 0.7928 - val_loss: 0.4256 - val_accuracy: 0.7648
Epoch 8/200
22014/22014 [=====] - 0s 19us/step - loss: 0.3440 - accuracy: 0.7949 - val_loss: 0.3811 - val_accuracy: 0.7937
Epoch 9/200
22014/22014 [=====] - 1s 24us/step - loss: 0.3395 - accuracy: 0.8036 - val_loss: 0.4546 - val_accuracy: 0.7427
Epoch 10/200
22014/22014 [=====] - 0s 19us/step - loss: 0.3365 - accuracy: 0.7998 - val_loss: 0.3892 - val_accuracy: 0.7851
Epoch 11/200
22014/22014 [=====] - 1s 26us/step - loss: 0.3296 - accuracy: 0.8037 - val_loss: 0.3675 - val_accuracy: 0.8017
Epoch 12/200
22014/22014 [=====] - 0s 19us/step - loss: 0.3257 - accuracy: 0.8037 - val_loss: 0.3606 - val_accuracy: 0.8241
Epoch 13/200
22014/22014 [=====] - 0s 22us/step - loss: 0.3207 - accuracy: 0.8123 - val_loss: 0.4424 - val_accuracy: 0.7829
Epoch 14/200
22014/22014 [=====] - 0s 18us/step - loss: 0.3180 - accuracy: 0.8108 - val_loss: 0.3670 - val_accuracy: 0.8133
Epoch 15/200
22014/22014 [=====] - 1s 27us/step - loss: 0.3115 - accuracy: 0.8142 - val_loss: 0.3409 - val_accuracy: 0.8384
Epoch 16/200
22014/22014 [=====] - 0s 22us/step - loss: 0.3076 - accuracy: 0.8145 - val_loss: 0.5024 - val_accuracy: 0.7363
Epoch 17/200
22014/22014 [=====] - 0s 22us/step - loss: 0.3045 - accuracy: 0.8230 - val_loss: 0.4379 - val_accuracy: 0.7637
Epoch 18/200
22014/22014 [=====] - 0s 22us/step - loss: 0.3011 - accuracy: 0.8187 - val_loss: 0.4581 - val_accuracy: 0.7823
Epoch 19/200
22014/22014 [=====] - 0s 19us/step - loss: 0.2957 - accuracy: 0.8220 - val_loss: 0.4728 - val_accuracy: 0.7664
Epoch 20/200
22014/22014 [=====] - 1s 31us/step - loss: 0.2962 - accuracy: 0.8203 - val_loss: 0.4315 - val_accuracy: 0.8041
Epoch 21/200

22014/22014 [=====] - 1s 25us/step - loss: 0.2912 - accuracy: 0.8262 - val_loss: 0.4079 - val_accuracy: 0.8025
Epoch 22/200
22014/22014 [=====] - 1s 27us/step - loss: 0.2871 - accuracy: 0.8289 - val_loss: 0.4407 - val_accuracy: 0.7860
Epoch 23/200
22014/22014 [=====] - 0s 21us/step - loss: 0.2864 - accuracy: 0.8289 - val_loss: 0.4557 - val_accuracy: 0.7925
Epoch 24/200
22014/22014 [=====] - 1s 25us/step - loss: 0.2802 - accuracy: 0.8337 - val_loss: 0.4789 - val_accuracy: 0.7832
Epoch 25/200
22014/22014 [=====] - 0s 20us/step - loss: 0.2800 - accuracy: 0.8337 - val_loss: 0.4538 - val_accuracy: 0.7949
Epoch 26/200
22014/22014 [=====] - 1s 25us/step - loss: 0.2801 - accuracy: 0.8272 - val_loss: 0.4999 - val_accuracy: 0.7708
Epoch 27/200
22014/22014 [=====] - 0s 18us/step - loss: 0.2751 - accuracy: 0.8317 - val_loss: 0.4437 - val_accuracy: 0.8085
Epoch 28/200
22014/22014 [=====] - 0s 21us/step - loss: 0.2727 - accuracy: 0.8372 - val_loss: 0.3851 - val_accuracy: 0.8060
Epoch 29/200
22014/22014 [=====] - 0s 21us/step - loss: 0.2691 - accuracy: 0.8370 - val_loss: 0.4437 - val_accuracy: 0.8103
Epoch 30/200
22014/22014 [=====] - 0s 23us/step - loss: 0.2691 - accuracy: 0.8382 - val_loss: 0.4539 - val_accuracy: 0.8259
Epoch 31/200
22014/22014 [=====] - 0s 20us/step - loss: 0.2701 - accuracy: 0.8358 - val_loss: 0.4987 - val_accuracy: 0.7907
Epoch 32/200
22014/22014 [=====] - 0s 18us/step - loss: 0.2643 - accuracy: 0.8401 - val_loss: 0.5038 - val_accuracy: 0.7956
Epoch 33/200
22014/22014 [=====] - 1s 27us/step - loss: 0.2604 - accuracy: 0.8417 - val_loss: 0.5550 - val_accuracy: 0.7684
Epoch 34/200
22014/22014 [=====] - 0s 18us/step - loss: 0.2609 - accuracy: 0.8373 - val_loss: 0.4585 - val_accuracy: 0.7945
Epoch 35/200
22014/22014 [=====] - 0s 22us/step - loss: 0.2572 - accuracy: 0.8396 - val_loss: 0.4660 - val_accuracy: 0.8284
Epoch 36/200
22014/22014 [=====] - 1s 23us/step - loss: 0.2561 - accuracy: 0.8422 - val_loss: 0.5567 - val_accuracy: 0.7745
Epoch 37/200

22014/22014 [=====] - 0s 22us/step - loss: 0.2610 - accuracy: 0.8407 - val_loss: 0.4992 - val_accuracy: 0.8092
Epoch 38/200
22014/22014 [=====] - 0s 22us/step - loss: 0.2540 - accuracy: 0.8412 - val_loss: 0.5585 - val_accuracy: 0.7457
Epoch 39/200
22014/22014 [=====] - 1s 28us/step - loss: 0.2557 - accuracy: 0.8377 - val_loss: 0.5159 - val_accuracy: 0.8053
Epoch 40/200
22014/22014 [=====] - 1s 23us/step - loss: 0.2466 - accuracy: 0.8441 - val_loss: 0.5017 - val_accuracy: 0.8120
Epoch 41/200
22014/22014 [=====] - 1s 28us/step - loss: 0.2543 - accuracy: 0.8434 - val_loss: 0.5718 - val_accuracy: 0.7164
Epoch 42/200
22014/22014 [=====] - 0s 19us/step - loss: 0.2439 - accuracy: 0.8463 - val_loss: 0.5459 - val_accuracy: 0.8123
Epoch 43/200
22014/22014 [=====] - 1s 23us/step - loss: 0.2458 - accuracy: 0.8433 - val_loss: 0.5167 - val_accuracy: 0.7988
Epoch 44/200
22014/22014 [=====] - 0s 22us/step - loss: 0.2512 - accuracy: 0.8440 - val_loss: 0.5358 - val_accuracy: 0.7977
Epoch 45/200
22014/22014 [=====] - 0s 20us/step - loss: 0.2386 - accuracy: 0.8444 - val_loss: 0.5027 - val_accuracy: 0.8273
Epoch 46/200
22014/22014 [=====] - 0s 21us/step - loss: 0.2386 - accuracy: 0.8482 - val_loss: 0.6107 - val_accuracy: 0.8060
Epoch 47/200
22014/22014 [=====] - 0s 18us/step - loss: 0.2428 - accuracy: 0.8425 - val_loss: 0.5180 - val_accuracy: 0.8385
Epoch 48/200
22014/22014 [=====] - 0s 22us/step - loss: 0.2438 - accuracy: 0.8463 - val_loss: 0.5627 - val_accuracy: 0.8253
Epoch 49/200
22014/22014 [=====] - 0s 19us/step - loss: 0.2403 - accuracy: 0.8505 - val_loss: 0.5544 - val_accuracy: 0.7911
Epoch 50/200
22014/22014 [=====] - 1s 24us/step - loss: 0.2322 - accuracy: 0.8517 - val_loss: 0.6342 - val_accuracy: 0.7479
Epoch 51/200
22014/22014 [=====] - 0s 20us/step - loss: 0.2353 - accuracy: 0.8520 - val_loss: 0.5482 - val_accuracy: 0.8179
Epoch 52/200
22014/22014 [=====] - 1s 24us/step - loss: 0.2387 - accuracy: 0.8496 - val_loss: 0.5543 - val_accuracy: 0.8064
Epoch 53/200

22014/22014 [=====] - 0s 19us/step - loss: 0.2352 -
 accuracy: 0.8529 - val_loss: 0.5451 - val_accuracy: 0.8051
 Epoch 54/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2391 -
 accuracy: 0.8491 - val_loss: 0.4896 - val_accuracy: 0.7828
 Epoch 55/200
 22014/22014 [=====] - 0s 18us/step - loss: 0.2328 -
 accuracy: 0.8506 - val_loss: 0.5507 - val_accuracy: 0.8023
 Epoch 56/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2304 -
 accuracy: 0.8495 - val_loss: 0.6483 - val_accuracy: 0.8071
 Epoch 57/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2286 -
 accuracy: 0.8519 - val_loss: 0.5910 - val_accuracy: 0.8007
 Epoch 58/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2267 -
 accuracy: 0.8515 - val_loss: 0.5778 - val_accuracy: 0.8279
 Epoch 59/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2276 -
 accuracy: 0.8567 - val_loss: 0.5555 - val_accuracy: 0.8185
 Epoch 60/200
 22014/22014 [=====] - 1s 25us/step - loss: 0.2304 -
 accuracy: 0.8525 - val_loss: 0.5807 - val_accuracy: 0.7692
 Epoch 61/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2193 -
 accuracy: 0.8619 - val_loss: 0.6509 - val_accuracy: 0.8292
 Epoch 62/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2357 -
 accuracy: 0.8563 - val_loss: 0.6189 - val_accuracy: 0.7835
 Epoch 63/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2267 -
 accuracy: 0.8579 - val_loss: 0.6743 - val_accuracy: 0.7991
 Epoch 64/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2245 -
 accuracy: 0.8557 - val_loss: 0.6307 - val_accuracy: 0.8015
 Epoch 65/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2270 -
 accuracy: 0.8511 - val_loss: 0.6844 - val_accuracy: 0.8065
 Epoch 66/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.2176 -
 accuracy: 0.8593 - val_loss: 0.6401 - val_accuracy: 0.8216
 Epoch 67/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2214 -
 accuracy: 0.8601 - val_loss: 0.7039 - val_accuracy: 0.7751
 Epoch 68/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.2210 -
 accuracy: 0.8613 - val_loss: 0.6935 - val_accuracy: 0.7779
 Epoch 69/200

22014/22014 [=====] - 0s 21us/step - loss: 0.2178 -
 accuracy: 0.8609 - val_loss: 0.7528 - val_accuracy: 0.7229
 Epoch 70/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2215 -
 accuracy: 0.8595 - val_loss: 0.5657 - val_accuracy: 0.7969
 Epoch 71/200
 22014/22014 [=====] - 1s 27us/step - loss: 0.2206 -
 accuracy: 0.8594 - val_loss: 0.5951 - val_accuracy: 0.7997
 Epoch 72/200
 22014/22014 [=====] - 1s 25us/step - loss: 0.2216 -
 accuracy: 0.8623 - val_loss: 0.6047 - val_accuracy: 0.7903
 Epoch 73/200
 22014/22014 [=====] - 1s 25us/step - loss: 0.2146 -
 accuracy: 0.8616 - val_loss: 0.6306 - val_accuracy: 0.8017
 Epoch 74/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2195 -
 accuracy: 0.8601 - val_loss: 0.5957 - val_accuracy: 0.8221
 Epoch 75/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2167 -
 accuracy: 0.8642 - val_loss: 0.7052 - val_accuracy: 0.8208
 Epoch 76/200
 22014/22014 [=====] - 0s 23us/step - loss: 0.2168 -
 accuracy: 0.8600 - val_loss: 0.6169 - val_accuracy: 0.8087
 Epoch 77/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2114 -
 accuracy: 0.8638 - val_loss: 0.5551 - val_accuracy: 0.7921
 Epoch 78/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.2152 -
 accuracy: 0.8636 - val_loss: 0.7109 - val_accuracy: 0.8297
 Epoch 79/200
 22014/22014 [=====] - 1s 27us/step - loss: 0.2141 -
 accuracy: 0.8679 - val_loss: 0.7735 - val_accuracy: 0.8276
 Epoch 80/200
 22014/22014 [=====] - 1s 27us/step - loss: 0.2160 -
 accuracy: 0.8630 - val_loss: 0.6073 - val_accuracy: 0.7808
 Epoch 81/200
 22014/22014 [=====] - 1s 25us/step - loss: 0.2120 -
 accuracy: 0.8649 - val_loss: 0.6369 - val_accuracy: 0.8161
 Epoch 82/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2099 -
 accuracy: 0.8645 - val_loss: 0.6713 - val_accuracy: 0.7936
 Epoch 83/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2134 -
 accuracy: 0.8696 - val_loss: 0.6212 - val_accuracy: 0.8279
 Epoch 84/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2077 -
 accuracy: 0.8704 - val_loss: 0.7087 - val_accuracy: 0.7571
 Epoch 85/200

22014/22014 [=====] - 0s 19us/step - loss: 0.2115 -
 accuracy: 0.8678 - val_loss: 0.6853 - val_accuracy: 0.8177
 Epoch 86/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2073 -
 accuracy: 0.8709 - val_loss: 0.7718 - val_accuracy: 0.8261
 Epoch 87/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.2104 -
 accuracy: 0.8706 - val_loss: 0.7053 - val_accuracy: 0.8185
 Epoch 88/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2035 -
 accuracy: 0.8686 - val_loss: 0.7563 - val_accuracy: 0.8111
 Epoch 89/200
 22014/22014 [=====] - 0s 18us/step - loss: 0.2088 -
 accuracy: 0.8665 - val_loss: 0.7043 - val_accuracy: 0.8352
 Epoch 90/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2019 -
 accuracy: 0.8699 - val_loss: 0.7195 - val_accuracy: 0.8133
 Epoch 91/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2143 -
 accuracy: 0.8728 - val_loss: 0.7261 - val_accuracy: 0.7545
 Epoch 92/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1995 -
 accuracy: 0.8731 - val_loss: 0.7529 - val_accuracy: 0.7733
 Epoch 93/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2056 -
 accuracy: 0.8699 - val_loss: 0.6976 - val_accuracy: 0.8057
 Epoch 94/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2034 -
 accuracy: 0.8714 - val_loss: 0.6837 - val_accuracy: 0.8332
 Epoch 95/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2009 -
 accuracy: 0.8744 - val_loss: 0.8735 - val_accuracy: 0.8055
 Epoch 96/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2066 -
 accuracy: 0.8690 - val_loss: 0.7801 - val_accuracy: 0.8356
 Epoch 97/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1940 -
 accuracy: 0.8769 - val_loss: 0.9140 - val_accuracy: 0.7484
 Epoch 98/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.2116 -
 accuracy: 0.8713 - val_loss: 0.8395 - val_accuracy: 0.8193
 Epoch 99/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1971 -
 accuracy: 0.8753 - val_loss: 0.8201 - val_accuracy: 0.8175
 Epoch 100/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1987 -
 accuracy: 0.8733 - val_loss: 0.7796 - val_accuracy: 0.8195
 Epoch 101/200

22014/22014 [=====] - 0s 21us/step - loss: 0.2019 -
 accuracy: 0.8755 - val_loss: 0.7396 - val_accuracy: 0.7957
 Epoch 102/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1969 -
 accuracy: 0.8714 - val_loss: 0.6214 - val_accuracy: 0.8128
 Epoch 103/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1928 -
 accuracy: 0.8761 - val_loss: 0.8249 - val_accuracy: 0.8052
 Epoch 104/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1915 -
 accuracy: 0.8772 - val_loss: 0.7459 - val_accuracy: 0.7755
 Epoch 105/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2020 -
 accuracy: 0.8710 - val_loss: 0.9243 - val_accuracy: 0.8268
 Epoch 106/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.1946 -
 accuracy: 0.8751 - val_loss: 0.9196 - val_accuracy: 0.8063
 Epoch 107/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1957 -
 accuracy: 0.8737 - val_loss: 0.9394 - val_accuracy: 0.8217
 Epoch 108/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1983 -
 accuracy: 0.8740 - val_loss: 0.7852 - val_accuracy: 0.8044
 Epoch 109/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1896 -
 accuracy: 0.8784 - val_loss: 0.8840 - val_accuracy: 0.7992
 Epoch 110/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2044 -
 accuracy: 0.8760 - val_loss: 0.8405 - val_accuracy: 0.8147
 Epoch 111/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1922 -
 accuracy: 0.8798 - val_loss: 0.9854 - val_accuracy: 0.8089
 Epoch 112/200
 22014/22014 [=====] - 1s 26us/step - loss: 0.1927 -
 accuracy: 0.8786 - val_loss: 0.8998 - val_accuracy: 0.8213
 Epoch 113/200
 22014/22014 [=====] - 1s 30us/step - loss: 0.1885 -
 accuracy: 0.8800 - val_loss: 0.8995 - val_accuracy: 0.8147
 Epoch 114/200
 22014/22014 [=====] - 1s 28us/step - loss: 0.1980 -
 accuracy: 0.8755 - val_loss: 0.8866 - val_accuracy: 0.8155
 Epoch 115/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1962 -
 accuracy: 0.8801 - val_loss: 0.9217 - val_accuracy: 0.7601
 Epoch 116/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1903 -
 accuracy: 0.8842 - val_loss: 0.8528 - val_accuracy: 0.8108
 Epoch 117/200

22014/22014 [=====] - 1s 26us/step - loss: 0.1909 -
 accuracy: 0.8798 - val_loss: 0.8652 - val_accuracy: 0.8376
 Epoch 118/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1976 -
 accuracy: 0.8811 - val_loss: 0.7272 - val_accuracy: 0.7989
 Epoch 119/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1838 -
 accuracy: 0.8843 - val_loss: 1.0096 - val_accuracy: 0.8259
 Epoch 120/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1986 -
 accuracy: 0.8793 - val_loss: 0.9809 - val_accuracy: 0.8049
 Epoch 121/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1845 -
 accuracy: 0.8825 - val_loss: 0.9231 - val_accuracy: 0.8037
 Epoch 122/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1879 -
 accuracy: 0.8849 - val_loss: 0.8919 - val_accuracy: 0.8184
 Epoch 123/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1870 -
 accuracy: 0.8830 - val_loss: 0.9810 - val_accuracy: 0.8232
 Epoch 124/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1840 -
 accuracy: 0.8858 - val_loss: 0.9030 - val_accuracy: 0.7996
 Epoch 125/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2079 -
 accuracy: 0.8774 - val_loss: 0.7612 - val_accuracy: 0.8299
 Epoch 126/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1823 -
 accuracy: 0.8862 - val_loss: 0.8475 - val_accuracy: 0.8197
 Epoch 127/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.1936 -
 accuracy: 0.8816 - val_loss: 0.8624 - val_accuracy: 0.7812
 Epoch 128/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1908 -
 accuracy: 0.8849 - val_loss: 0.7172 - val_accuracy: 0.8064
 Epoch 129/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1858 -
 accuracy: 0.8831 - val_loss: 0.7372 - val_accuracy: 0.8227
 Epoch 130/200
 22014/22014 [=====] - 0s 18us/step - loss: 0.1916 -
 accuracy: 0.8816 - val_loss: 0.8725 - val_accuracy: 0.8023
 Epoch 131/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.2116 -
 accuracy: 0.8872 - val_loss: 0.9028 - val_accuracy: 0.8272
 Epoch 132/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1779 -
 accuracy: 0.8865 - val_loss: 0.8736 - val_accuracy: 0.8312
 Epoch 133/200

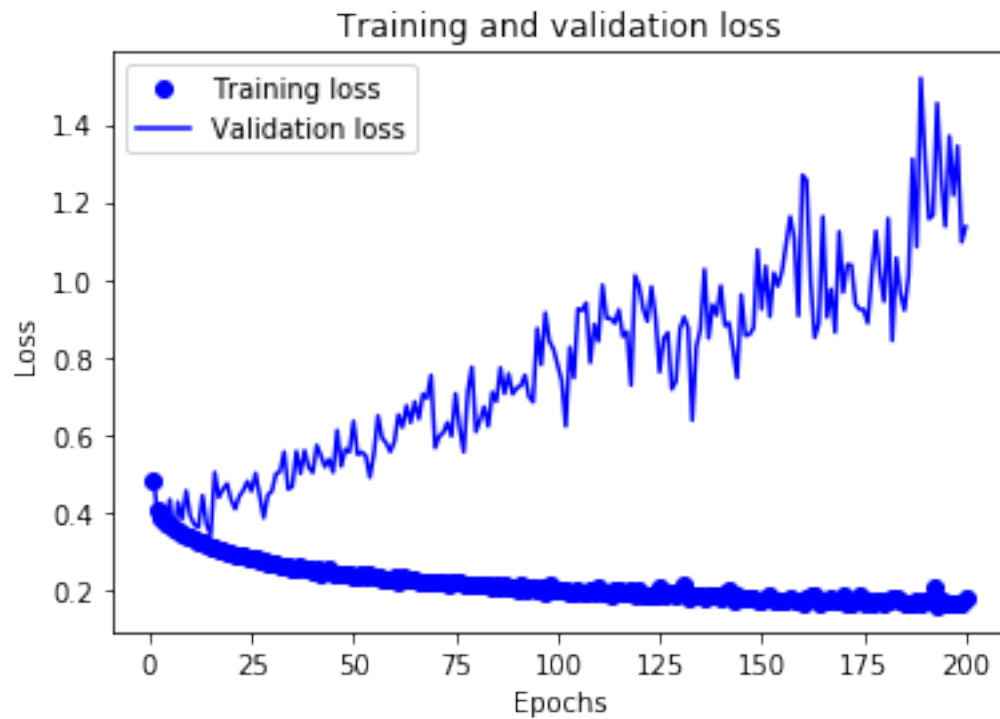
22014/22014 [=====] - 1s 23us/step - loss: 0.1888 - accuracy: 0.8859 - val_loss: 0.6364 - val_accuracy: 0.8437
Epoch 134/200
22014/22014 [=====] - 1s 23us/step - loss: 0.1821 - accuracy: 0.8896 - val_loss: 0.8263 - val_accuracy: 0.8241
Epoch 135/200
22014/22014 [=====] - 0s 21us/step - loss: 0.1830 - accuracy: 0.8885 - val_loss: 0.8715 - val_accuracy: 0.8251
Epoch 136/200
22014/22014 [=====] - 0s 22us/step - loss: 0.1772 - accuracy: 0.8895 - val_loss: 1.0257 - val_accuracy: 0.8317
Epoch 137/200
22014/22014 [=====] - 1s 23us/step - loss: 0.1850 - accuracy: 0.8869 - val_loss: 0.8483 - val_accuracy: 0.8131
Epoch 138/200
22014/22014 [=====] - 1s 23us/step - loss: 0.1839 - accuracy: 0.8878 - val_loss: 0.9348 - val_accuracy: 0.8200
Epoch 139/200
22014/22014 [=====] - 0s 21us/step - loss: 0.1831 - accuracy: 0.8870 - val_loss: 0.9079 - val_accuracy: 0.8388
Epoch 140/200
22014/22014 [=====] - 0s 22us/step - loss: 0.1762 - accuracy: 0.8887 - val_loss: 0.9835 - val_accuracy: 0.8175
Epoch 141/200
22014/22014 [=====] - 1s 24us/step - loss: 0.1926 - accuracy: 0.8868 - val_loss: 0.8806 - val_accuracy: 0.8227
Epoch 142/200
22014/22014 [=====] - 0s 23us/step - loss: 0.1961 - accuracy: 0.8903 - val_loss: 0.8878 - val_accuracy: 0.8000
Epoch 143/200
22014/22014 [=====] - 0s 21us/step - loss: 0.1736 - accuracy: 0.8928 - val_loss: 0.8197 - val_accuracy: 0.7947
Epoch 144/200
22014/22014 [=====] - 1s 24us/step - loss: 0.1823 - accuracy: 0.8903 - val_loss: 0.7459 - val_accuracy: 0.7680
Epoch 145/200
22014/22014 [=====] - 1s 29us/step - loss: 0.1799 - accuracy: 0.8923 - val_loss: 0.9606 - val_accuracy: 0.8044
Epoch 146/200
22014/22014 [=====] - 1s 25us/step - loss: 0.1803 - accuracy: 0.8879 - val_loss: 0.8550 - val_accuracy: 0.8233
Epoch 147/200
22014/22014 [=====] - 1s 23us/step - loss: 0.1784 - accuracy: 0.8903 - val_loss: 0.8576 - val_accuracy: 0.8315
Epoch 148/200
22014/22014 [=====] - 0s 20us/step - loss: 0.1809 - accuracy: 0.8929 - val_loss: 0.8744 - val_accuracy: 0.8276
Epoch 149/200

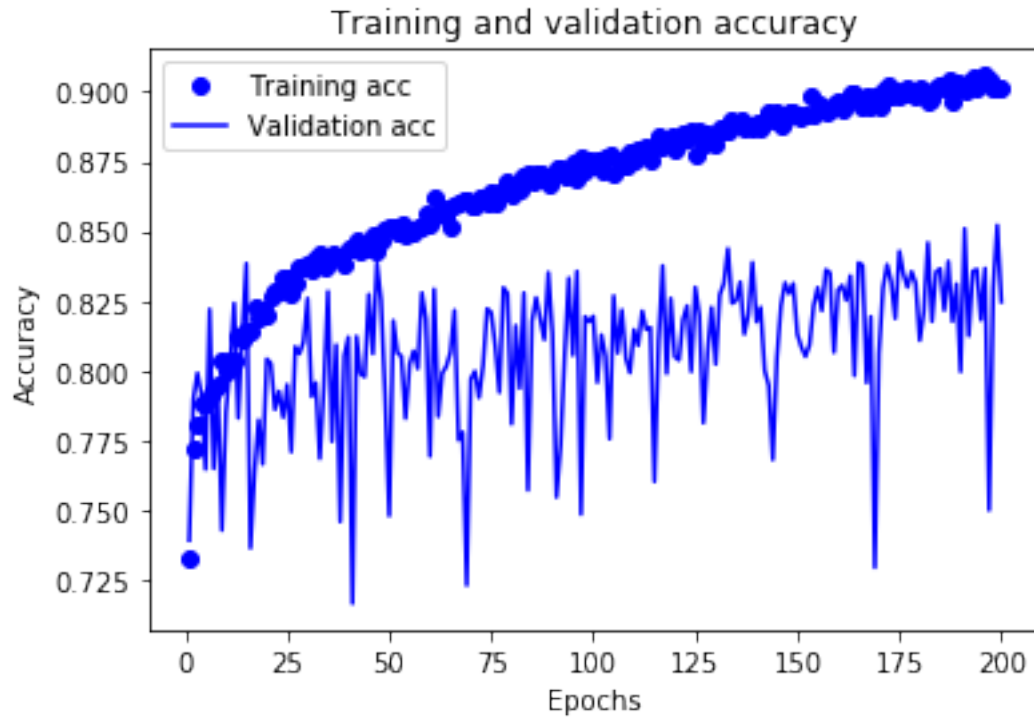
22014/22014 [=====] - 0s 23us/step - loss: 0.1747 -
 accuracy: 0.8927 - val_loss: 1.0759 - val_accuracy: 0.8312
 Epoch 150/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.1747 -
 accuracy: 0.8916 - val_loss: 0.9231 - val_accuracy: 0.8131
 Epoch 151/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1774 -
 accuracy: 0.8912 - val_loss: 1.0339 - val_accuracy: 0.8084
 Epoch 152/200
 22014/22014 [=====] - 0s 19us/step - loss: 0.1836 -
 accuracy: 0.8922 - val_loss: 0.9041 - val_accuracy: 0.8051
 Epoch 153/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1718 -
 accuracy: 0.8987 - val_loss: 1.0167 - val_accuracy: 0.8092
 Epoch 154/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1745 -
 accuracy: 0.8918 - val_loss: 0.9828 - val_accuracy: 0.8233
 Epoch 155/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1713 -
 accuracy: 0.8956 - val_loss: 1.0213 - val_accuracy: 0.8297
 Epoch 156/200
 22014/22014 [=====] - 0s 18us/step - loss: 0.1730 -
 accuracy: 0.8952 - val_loss: 1.0914 - val_accuracy: 0.8215
 Epoch 157/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1772 -
 accuracy: 0.8927 - val_loss: 1.1620 - val_accuracy: 0.8361
 Epoch 158/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1743 -
 accuracy: 0.8940 - val_loss: 1.1050 - val_accuracy: 0.8351
 Epoch 159/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1738 -
 accuracy: 0.8956 - val_loss: 0.9062 - val_accuracy: 0.8065
 Epoch 160/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1680 -
 accuracy: 0.8963 - val_loss: 1.2690 - val_accuracy: 0.8289
 Epoch 161/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1813 -
 accuracy: 0.8935 - val_loss: 1.2557 - val_accuracy: 0.8305
 Epoch 162/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1876 -
 accuracy: 0.8970 - val_loss: 0.9954 - val_accuracy: 0.8252
 Epoch 163/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1858 -
 accuracy: 0.8993 - val_loss: 0.8500 - val_accuracy: 0.8340
 Epoch 164/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1660 -
 accuracy: 0.8997 - val_loss: 0.8859 - val_accuracy: 0.7981
 Epoch 165/200

22014/22014 [=====] - 1s 24us/step - loss: 0.1695 -
 accuracy: 0.8974 - val_loss: 1.1621 - val_accuracy: 0.8387
 Epoch 166/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1733 -
 accuracy: 0.8946 - val_loss: 0.9025 - val_accuracy: 0.8377
 Epoch 167/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1763 -
 accuracy: 0.8989 - val_loss: 0.9745 - val_accuracy: 0.7956
 Epoch 168/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1708 -
 accuracy: 0.8950 - val_loss: 0.8637 - val_accuracy: 0.8193
 Epoch 169/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1732 -
 accuracy: 0.8981 - val_loss: 1.1227 - val_accuracy: 0.7293
 Epoch 170/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1874 -
 accuracy: 0.8944 - val_loss: 0.9670 - val_accuracy: 0.8061
 Epoch 171/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1675 -
 accuracy: 0.9002 - val_loss: 1.0383 - val_accuracy: 0.8296
 Epoch 172/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1648 -
 accuracy: 0.9027 - val_loss: 1.0367 - val_accuracy: 0.8380
 Epoch 173/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1701 -
 accuracy: 0.8989 - val_loss: 0.9359 - val_accuracy: 0.8319
 Epoch 174/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1865 -
 accuracy: 0.9003 - val_loss: 0.9227 - val_accuracy: 0.8187
 Epoch 175/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1668 -
 accuracy: 0.8986 - val_loss: 0.9218 - val_accuracy: 0.8428
 Epoch 176/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1743 -
 accuracy: 0.8990 - val_loss: 0.8874 - val_accuracy: 0.8347
 Epoch 177/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1652 -
 accuracy: 0.9011 - val_loss: 1.0134 - val_accuracy: 0.8299
 Epoch 178/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1691 -
 accuracy: 0.9003 - val_loss: 1.1242 - val_accuracy: 0.8355
 Epoch 179/200
 22014/22014 [=====] - 1s 25us/step - loss: 0.1733 -
 accuracy: 0.8986 - val_loss: 1.0155 - val_accuracy: 0.8313
 Epoch 180/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1655 -
 accuracy: 0.9011 - val_loss: 0.9432 - val_accuracy: 0.8108
 Epoch 181/200

22014/22014 [=====] - 1s 24us/step - loss: 0.1704 -
 accuracy: 0.9000 - val_loss: 1.1567 - val_accuracy: 0.8212
 Epoch 182/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1764 -
 accuracy: 0.8969 - val_loss: 0.8420 - val_accuracy: 0.8459
 Epoch 183/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1770 -
 accuracy: 0.8977 - val_loss: 1.0550 - val_accuracy: 0.8176
 Epoch 184/200
 22014/22014 [=====] - 1s 28us/step - loss: 0.1689 -
 accuracy: 0.9010 - val_loss: 0.9633 - val_accuracy: 0.8353
 Epoch 185/200
 22014/22014 [=====] - 1s 26us/step - loss: 0.1642 -
 accuracy: 0.9028 - val_loss: 0.9206 - val_accuracy: 0.8365
 Epoch 186/200
 22014/22014 [=====] - 1s 26us/step - loss: 0.1672 -
 accuracy: 0.9015 - val_loss: 1.0095 - val_accuracy: 0.8216
 Epoch 187/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1631 -
 accuracy: 0.9038 - val_loss: 1.3106 - val_accuracy: 0.8392
 Epoch 188/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1738 -
 accuracy: 0.8968 - val_loss: 1.0838 - val_accuracy: 0.8176
 Epoch 189/200
 22014/22014 [=====] - 0s 20us/step - loss: 0.1626 -
 accuracy: 0.9030 - val_loss: 1.5192 - val_accuracy: 0.8311
 Epoch 190/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1686 -
 accuracy: 0.9015 - val_loss: 1.3179 - val_accuracy: 0.7996
 Epoch 191/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.1697 -
 accuracy: 0.9009 - val_loss: 1.1561 - val_accuracy: 0.8509
 Epoch 192/200
 22014/22014 [=====] - 0s 21us/step - loss: 0.2058 -
 accuracy: 0.9029 - val_loss: 1.1658 - val_accuracy: 0.8124
 Epoch 193/200
 22014/22014 [=====] - 1s 24us/step - loss: 0.1584 -
 accuracy: 0.9050 - val_loss: 1.4541 - val_accuracy: 0.8356
 Epoch 194/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1744 -
 accuracy: 0.9013 - val_loss: 1.2607 - val_accuracy: 0.8363
 Epoch 195/200
 22014/22014 [=====] - 0s 22us/step - loss: 0.1653 -
 accuracy: 0.9030 - val_loss: 1.1369 - val_accuracy: 0.8181
 Epoch 196/200
 22014/22014 [=====] - 1s 23us/step - loss: 0.1626 -
 accuracy: 0.9060 - val_loss: 1.3697 - val_accuracy: 0.8367
 Epoch 197/200

22014/22014 [=====] - 0s 21us/step - loss: 0.1680 -
accuracy: 0.9042 - val_loss: 1.2173 - val_accuracy: 0.7499
Epoch 198/200
22014/22014 [=====] - 1s 24us/step - loss: 0.1620 -
accuracy: 0.9015 - val_loss: 1.3438 - val_accuracy: 0.8296
Epoch 199/200
22014/22014 [=====] - 0s 22us/step - loss: 0.1645 -
accuracy: 0.9016 - val_loss: 1.0962 - val_accuracy: 0.8521
Epoch 200/200
22014/22014 [=====] - 1s 24us/step - loss: 0.1753 -
accuracy: 0.9016 - val_loss: 1.1354 - val_accuracy: 0.8247





```

confusion matrix
[[5568 1046]
 [ 269  617]]
accuracy = 0.8246666666666667
precision = 0.3710162357185809
recall = 0.6963882618510158
F1 score = 0.4841114162416634

```

2.4 Deal with overfitting

Our model has already performed well on our training data.

Now, let's use 'dropout' method to deal with the overfitting problem.

```

[45]: model = models.Sequential()
      model.add(layers.Dense(100, activation='relu'))
      model.add(layers.Dropout(0.1))
      model.add(layers.Dense(100, activation='relu'))
      model.add(layers.Dropout(0.3))
      model.add(layers.Dense(100, activation='relu'))
      model.add(layers.Dropout(0.3))
      model.add(layers.Dense(100, activation='relu'))
      model.add(layers.Dropout(0.3))
      model.add(layers.Dense(100, activation='relu'))

```

```

model.add(layers.Dropout(0.3))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.1))
model.add(layers.Dense(1, activation='sigmoid'))

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

history = model.fit(partial_X_train,
                    partial_y_train,
                    epochs=50,
                    batch_size=512,
                    validation_data=(X_valid, y_valid),
                    class_weight=class_weights)

# plot the results of loss values from the training set and validation set
history_dict = history.history
loss_values = history_dict['loss']
val_loss_values = history_dict['val_loss']

epochs = range(1, len(history_dict['accuracy']) + 1)

plt.plot(epochs, loss_values, 'bo', label='Training loss')
plt.plot(epochs, val_loss_values, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()

#plt.clf()
acc = history_dict['accuracy']
val_acc = history_dict['val_accuracy']
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()

# Compute the confusion matrix
y_preds = model.predict(X_valid)
y_preds = (y_preds>0.5).astype(int)
print('confusion matrix')
print(sklearn.metrics.confusion_matrix(y_valid,y_preds))

```

```

# Compute accuracy
print('accuracy = ' + str(sklearn.metrics.accuracy_score(y_valid,y_preds)))

# Compute Precision
print('precision = ' + str(sklearn.metrics.precision_score(y_valid,y_preds)))

# Compute Recall
print('recall = ' + str(sklearn.metrics.recall_score(y_valid,y_preds)))

# Compute F1 score
print('F1 score = ' + str(sklearn.metrics.f1_score(y_valid,y_preds)))

```

Train on 22014 samples, validate on 7500 samples

Epoch 1/50

22014/22014 [=====] - 2s 80us/step - loss: 0.5662 - accuracy: 0.6973 - val_loss: 0.4001 - val_accuracy: 0.7769

Epoch 2/50

22014/22014 [=====] - 1s 32us/step - loss: 0.4537 - accuracy: 0.7542 - val_loss: 0.3703 - val_accuracy: 0.7779

Epoch 3/50

22014/22014 [=====] - 1s 27us/step - loss: 0.4294 - accuracy: 0.7587 - val_loss: 0.4032 - val_accuracy: 0.7615

Epoch 4/50

22014/22014 [=====] - 1s 24us/step - loss: 0.4108 - accuracy: 0.7661 - val_loss: 0.3641 - val_accuracy: 0.7780

Epoch 5/50

22014/22014 [=====] - 1s 25us/step - loss: 0.4001 - accuracy: 0.7774 - val_loss: 0.4188 - val_accuracy: 0.7597

Epoch 6/50

22014/22014 [=====] - 1s 41us/step - loss: 0.3936 - accuracy: 0.7825 - val_loss: 0.3717 - val_accuracy: 0.7755

Epoch 7/50

22014/22014 [=====] - 1s 41us/step - loss: 0.3903 - accuracy: 0.7784 - val_loss: 0.3694 - val_accuracy: 0.7836

Epoch 8/50

22014/22014 [=====] - 1s 37us/step - loss: 0.3804 - accuracy: 0.7867 - val_loss: 0.3645 - val_accuracy: 0.7876

Epoch 9/50

22014/22014 [=====] - 1s 28us/step - loss: 0.3827 - accuracy: 0.7840 - val_loss: 0.3466 - val_accuracy: 0.7813

Epoch 10/50

22014/22014 [=====] - 1s 29us/step - loss: 0.3733 - accuracy: 0.7883 - val_loss: 0.3950 - val_accuracy: 0.7648

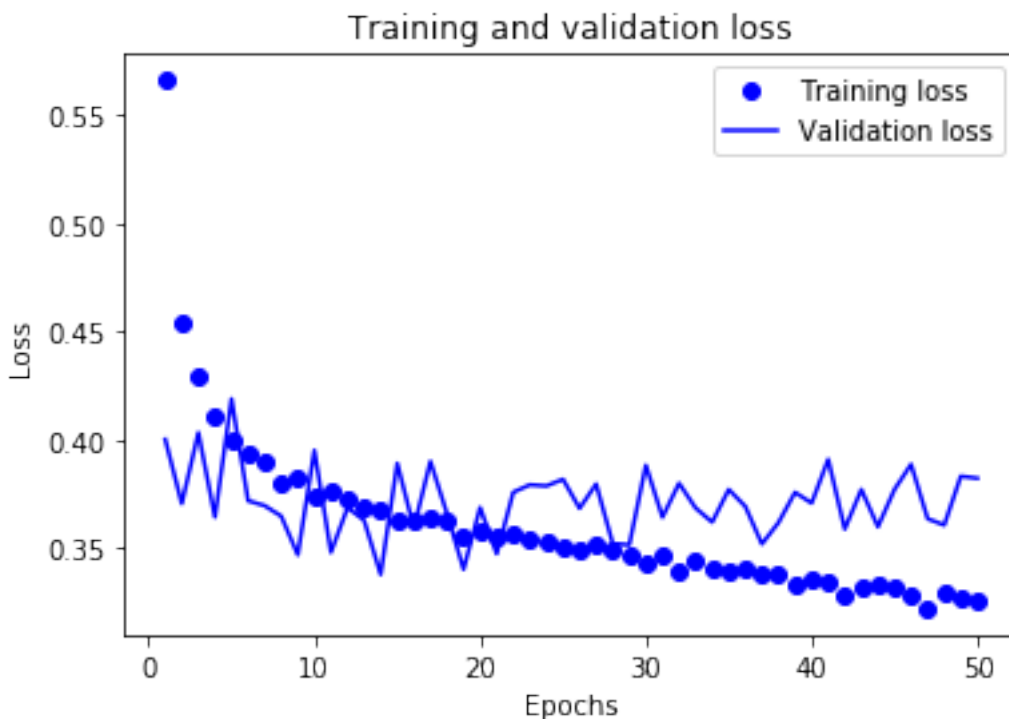
Epoch 11/50

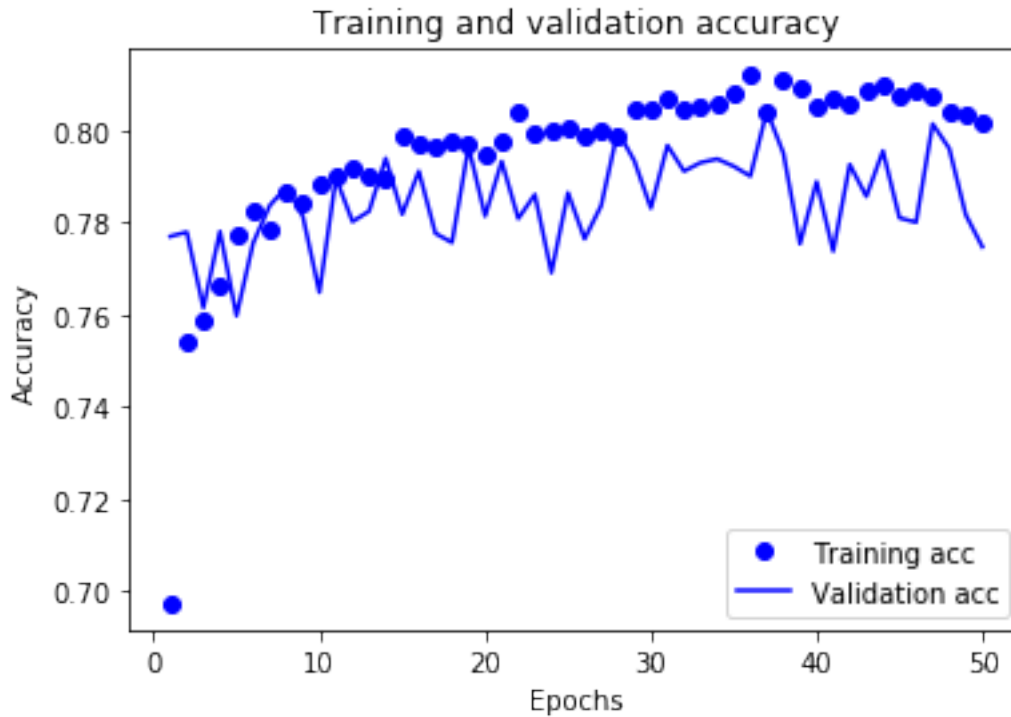
22014/22014 [=====] - 1s 28us/step - loss: 0.3756 -

accuracy: 0.7897 - val_loss: 0.3478 - val_accuracy: 0.7896
 Epoch 12/50
 22014/22014 [=====] - 1s 33us/step - loss: 0.3726 -
 accuracy: 0.7920 - val_loss: 0.3684 - val_accuracy: 0.7801
 Epoch 13/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3686 -
 accuracy: 0.7898 - val_loss: 0.3626 - val_accuracy: 0.7824
 Epoch 14/50
 22014/22014 [=====] - 1s 36us/step - loss: 0.3676 -
 accuracy: 0.7895 - val_loss: 0.3376 - val_accuracy: 0.7939
 Epoch 15/50
 22014/22014 [=====] - 1s 33us/step - loss: 0.3619 -
 accuracy: 0.7985 - val_loss: 0.3890 - val_accuracy: 0.7817
 Epoch 16/50
 22014/22014 [=====] - 1s 33us/step - loss: 0.3623 -
 accuracy: 0.7969 - val_loss: 0.3591 - val_accuracy: 0.7911
 Epoch 17/50
 22014/22014 [=====] - 1s 37us/step - loss: 0.3643 -
 accuracy: 0.7962 - val_loss: 0.3899 - val_accuracy: 0.7776
 Epoch 18/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3621 -
 accuracy: 0.7975 - val_loss: 0.3657 - val_accuracy: 0.7756
 Epoch 19/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3548 -
 accuracy: 0.7968 - val_loss: 0.3399 - val_accuracy: 0.7964
 Epoch 20/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3577 -
 accuracy: 0.7947 - val_loss: 0.3685 - val_accuracy: 0.7813
 Epoch 21/50
 22014/22014 [=====] - 1s 34us/step - loss: 0.3547 -
 accuracy: 0.7975 - val_loss: 0.3471 - val_accuracy: 0.7932
 Epoch 22/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3560 -
 accuracy: 0.8039 - val_loss: 0.3756 - val_accuracy: 0.7808
 Epoch 23/50
 22014/22014 [=====] - 1s 31us/step - loss: 0.3535 -
 accuracy: 0.7995 - val_loss: 0.3790 - val_accuracy: 0.7860
 Epoch 24/50
 22014/22014 [=====] - 1s 45us/step - loss: 0.3532 -
 accuracy: 0.8001 - val_loss: 0.3785 - val_accuracy: 0.7691
 Epoch 25/50
 22014/22014 [=====] - 1s 36us/step - loss: 0.3497 -
 accuracy: 0.8005 - val_loss: 0.3816 - val_accuracy: 0.7864
 Epoch 26/50
 22014/22014 [=====] - 1s 39us/step - loss: 0.3490 -
 accuracy: 0.7986 - val_loss: 0.3681 - val_accuracy: 0.7764
 Epoch 27/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3520 -

accuracy: 0.7997 - val_loss: 0.3794 - val_accuracy: 0.7836
 Epoch 28/50
 22014/22014 [=====] - 1s 29us/step - loss: 0.3485 -
 accuracy: 0.7987 - val_loss: 0.3518 - val_accuracy: 0.7997
 Epoch 29/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3460 -
 accuracy: 0.8046 - val_loss: 0.3513 - val_accuracy: 0.7933
 Epoch 30/50
 22014/22014 [=====] - 1s 31us/step - loss: 0.3425 -
 accuracy: 0.8047 - val_loss: 0.3881 - val_accuracy: 0.7831
 Epoch 31/50
 22014/22014 [=====] - 1s 28us/step - loss: 0.3469 -
 accuracy: 0.8065 - val_loss: 0.3640 - val_accuracy: 0.7967
 Epoch 32/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3391 -
 accuracy: 0.8045 - val_loss: 0.3798 - val_accuracy: 0.7911
 Epoch 33/50
 22014/22014 [=====] - 1s 32us/step - loss: 0.3441 -
 accuracy: 0.8053 - val_loss: 0.3683 - val_accuracy: 0.7929
 Epoch 34/50
 22014/22014 [=====] - 1s 34us/step - loss: 0.3404 -
 accuracy: 0.8057 - val_loss: 0.3618 - val_accuracy: 0.7937
 Epoch 35/50
 22014/22014 [=====] - 1s 36us/step - loss: 0.3396 -
 accuracy: 0.8079 - val_loss: 0.3769 - val_accuracy: 0.7921
 Epoch 36/50
 22014/22014 [=====] - 1s 29us/step - loss: 0.3408 -
 accuracy: 0.8119 - val_loss: 0.3690 - val_accuracy: 0.7900
 Epoch 37/50
 22014/22014 [=====] - 1s 26us/step - loss: 0.3381 -
 accuracy: 0.8040 - val_loss: 0.3517 - val_accuracy: 0.8041
 Epoch 38/50
 22014/22014 [=====] - 1s 27us/step - loss: 0.3382 -
 accuracy: 0.8108 - val_loss: 0.3615 - val_accuracy: 0.7949
 Epoch 39/50
 22014/22014 [=====] - 1s 27us/step - loss: 0.3326 -
 accuracy: 0.8093 - val_loss: 0.3755 - val_accuracy: 0.7753
 Epoch 40/50
 22014/22014 [=====] - 1s 28us/step - loss: 0.3359 -
 accuracy: 0.8050 - val_loss: 0.3706 - val_accuracy: 0.7888
 Epoch 41/50
 22014/22014 [=====] - 1s 26us/step - loss: 0.3342 -
 accuracy: 0.8069 - val_loss: 0.3908 - val_accuracy: 0.7737
 Epoch 42/50
 22014/22014 [=====] - 1s 25us/step - loss: 0.3285 -
 accuracy: 0.8055 - val_loss: 0.3585 - val_accuracy: 0.7925
 Epoch 43/50
 22014/22014 [=====] - 1s 28us/step - loss: 0.3318 -

accuracy: 0.8088 - val_loss: 0.3769 - val_accuracy: 0.7856
Epoch 44/50
22014/22014 [=====] - 1s 27us/step - loss: 0.3331 -
accuracy: 0.8098 - val_loss: 0.3596 - val_accuracy: 0.7955
Epoch 45/50
22014/22014 [=====] - 1s 29us/step - loss: 0.3317 -
accuracy: 0.8071 - val_loss: 0.3769 - val_accuracy: 0.7809
Epoch 46/50
22014/22014 [=====] - 1s 28us/step - loss: 0.3284 -
accuracy: 0.8088 - val_loss: 0.3885 - val_accuracy: 0.7800
Epoch 47/50
22014/22014 [=====] - 1s 28us/step - loss: 0.3218 -
accuracy: 0.8071 - val_loss: 0.3633 - val_accuracy: 0.8013
Epoch 48/50
22014/22014 [=====] - 1s 40us/step - loss: 0.3287 -
accuracy: 0.8039 - val_loss: 0.3604 - val_accuracy: 0.7960
Epoch 49/50
22014/22014 [=====] - 1s 32us/step - loss: 0.3272 -
accuracy: 0.8031 - val_loss: 0.3830 - val_accuracy: 0.7816
Epoch 50/50
22014/22014 [=====] - 1s 26us/step - loss: 0.3254 -
accuracy: 0.8016 - val_loss: 0.3821 - val_accuracy: 0.7747





```

confusion matrix
[[5049 1565]
 [ 125  761]]
accuracy = 0.7746666666666666
precision = 0.3271711092003439
recall = 0.8589164785553047
F1 score = 0.4738480697384807

```

2.5 Final DNN model

Now, I can use all the training data to train this model.

[47]: *# Use whole training data to train our final model*

```

model = models.Sequential()
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.1))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.3))

```

```

model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.3))
model.add(layers.Dense(100, activation='relu'))
model.add(layers.Dropout(0.1))
model.add(layers.Dense(1, activation='sigmoid'))

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

history = model.fit(X_train,
                    y_train,
                    epochs=50,
                    batch_size=512,
                    class_weight=class_weights)

```

```

Epoch 1/50
29514/29514 [=====] - 1s 48us/step - loss: 0.5404 -
accuracy: 0.7186
Epoch 2/50
29514/29514 [=====] - 1s 25us/step - loss: 0.4468 -
accuracy: 0.7585
Epoch 3/50
29514/29514 [=====] - 1s 24us/step - loss: 0.4242 -
accuracy: 0.7700
Epoch 4/50
29514/29514 [=====] - 1s 25us/step - loss: 0.4082 -
accuracy: 0.7787
Epoch 5/50
29514/29514 [=====] - 1s 25us/step - loss: 0.4013 -
accuracy: 0.7742
Epoch 6/50
29514/29514 [=====] - 1s 24us/step - loss: 0.3923 -
accuracy: 0.7889
Epoch 7/50
29514/29514 [=====] - 1s 27us/step - loss: 0.3889 -
accuracy: 0.7794
Epoch 8/50
29514/29514 [=====] - 1s 24us/step - loss: 0.3877 -
accuracy: 0.7906
Epoch 9/50
29514/29514 [=====] - 1s 22us/step - loss: 0.3823 -
accuracy: 0.7877
Epoch 10/50
29514/29514 [=====] - 1s 22us/step - loss: 0.3806 -
accuracy: 0.7870
Epoch 11/50

```



```

29514/29514 [=====] - 1s 23us/step - loss: 0.3799 -
accuracy: 0.7954
Epoch 12/50
29514/29514 [=====] - 1s 23us/step - loss: 0.3765 -
accuracy: 0.7884
Epoch 13/50
29514/29514 [=====] - 1s 24us/step - loss: 0.3738 -
accuracy: 0.7949
Epoch 14/50
29514/29514 [=====] - 1s 22us/step - loss: 0.3721 -
accuracy: 0.7986
Epoch 15/50
29514/29514 [=====] - 1s 22us/step - loss: 0.3686 -
accuracy: 0.7959
Epoch 16/50
29514/29514 [=====] - 1s 23us/step - loss: 0.3667 -
accuracy: 0.7979
Epoch 17/50
29514/29514 [=====] - 1s 25us/step - loss: 0.3663 -
accuracy: 0.7995
Epoch 18/50
29514/29514 [=====] - 1s 34us/step - loss: 0.3643 -
accuracy: 0.8021
Epoch 19/50
29514/29514 [=====] - 1s 28us/step - loss: 0.3603 -
accuracy: 0.8020
Epoch 20/50
29514/29514 [=====] - 1s 24us/step - loss: 0.3612 -
accuracy: 0.8091
Epoch 21/50
29514/29514 [=====] - 1s 23us/step - loss: 0.3589 -
accuracy: 0.8082
Epoch 22/50
29514/29514 [=====] - 1s 22us/step - loss: 0.3568 -
accuracy: 0.8107
Epoch 23/50
29514/29514 [=====] - 1s 25us/step - loss: 0.3570 -
accuracy: 0.8043
Epoch 24/50
29514/29514 [=====] - 1s 24us/step - loss: 0.3558 -
accuracy: 0.8106
Epoch 25/50
29514/29514 [=====] - 1s 35us/step - loss: 0.3533 -
accuracy: 0.8116 0s
Epoch 26/50
29514/29514 [=====] - 1s 34us/step - loss: 0.3523 -
accuracy: 0.8077
Epoch 27/50

```

29514/29514 [=====] - 1s 27us/step - loss: 0.3542 -
 accuracy: 0.8105
 Epoch 28/50
 29514/29514 [=====] - 1s 28us/step - loss: 0.3474 -
 accuracy: 0.8093
 Epoch 29/50
 29514/29514 [=====] - 1s 32us/step - loss: 0.3486 -
 accuracy: 0.8096
 Epoch 30/50
 29514/29514 [=====] - 1s 34us/step - loss: 0.3499 -
 accuracy: 0.8154
 Epoch 31/50
 29514/29514 [=====] - 1s 25us/step - loss: 0.3469 -
 accuracy: 0.8150
 Epoch 32/50
 29514/29514 [=====] - 1s 25us/step - loss: 0.3438 -
 accuracy: 0.8095
 Epoch 33/50
 29514/29514 [=====] - 1s 28us/step - loss: 0.3422 -
 accuracy: 0.8105
 Epoch 34/50
 29514/29514 [=====] - 1s 27us/step - loss: 0.3409 -
 accuracy: 0.8079
 Epoch 35/50
 29514/29514 [=====] - 1s 30us/step - loss: 0.3461 -
 accuracy: 0.8105
 Epoch 36/50
 29514/29514 [=====] - 1s 27us/step - loss: 0.3433 -
 accuracy: 0.8145
 Epoch 37/50
 29514/29514 [=====] - 1s 24us/step - loss: 0.3397 -
 accuracy: 0.8145
 Epoch 38/50
 29514/29514 [=====] - 1s 24us/step - loss: 0.3390 -
 accuracy: 0.8123
 Epoch 39/50
 29514/29514 [=====] - 1s 39us/step - loss: 0.3391 -
 accuracy: 0.8105
 Epoch 40/50
 29514/29514 [=====] - 1s 29us/step - loss: 0.3384 -
 accuracy: 0.8097
 Epoch 41/50
 29514/29514 [=====] - 1s 26us/step - loss: 0.3372 -
 accuracy: 0.8140
 Epoch 42/50
 29514/29514 [=====] - 1s 26us/step - loss: 0.3388 -
 accuracy: 0.8099
 Epoch 43/50

```

29514/29514 [=====] - 1s 26us/step - loss: 0.3367 -
accuracy: 0.8122
Epoch 44/50
29514/29514 [=====] - 1s 25us/step - loss: 0.3417 -
accuracy: 0.8095
Epoch 45/50
29514/29514 [=====] - 1s 23us/step - loss: 0.3355 -
accuracy: 0.8067
Epoch 46/50
29514/29514 [=====] - 1s 25us/step - loss: 0.3377 -
accuracy: 0.8112
Epoch 47/50
29514/29514 [=====] - 1s 23us/step - loss: 0.3339 -
accuracy: 0.8088
Epoch 48/50
29514/29514 [=====] - 1s 25us/step - loss: 0.3359 -
accuracy: 0.8056
Epoch 49/50
29514/29514 [=====] - 1s 30us/step - loss: 0.3333 -
accuracy: 0.8092
Epoch 50/50
29514/29514 [=====] - 1s 30us/step - loss: 0.3363 -
accuracy: 0.8029

```

After training our final model, we can then use this model to predict our final answer (use our test dataset).

2.6 Using a trained network to generate predictions on testing data

```

[49]: #Using a trained network to generate predictions on new data
y_preds=model.predict(X_test)
y_preds=(y_preds>0.5).astype(int)
answer=pd.DataFrame(y_preds)

X_test=pd.read_csv('/Users/Stylewsxcde991/Desktop/      /qbs-competition-1/data/
↳test.csv',index_col=0)
answer.index=X_test.index
result = pd.read_csv('/Users/Stylewsxcde991/Desktop/      /qbs-competition-1/
↳data/sub_try.csv',index_col=0)
result['Target'] = answer
result.to_csv('/Users/Stylewsxcde991/Desktop/      /qbs-competition-1/data/
↳Result_1.csv')

```

After some dataframe operations, we can then export our answer.

Now, let's see the prediction result of our final model.

2.7 The prediction result:

As we talked before, by using our final weighted model, in Kaggle competition, the public score of this model is about 0.83 (the private score is also about 0.82).

In my opinion, this model's great performance in Kaggle competition means this model can effectively predict whether a person makes over 50K a year. That is to say, if we want to predict if a person makes over 50K a year, our final model is trustable and reasonable.

2.8 Learning progress and reflection

To be honest, when I first built a NN model for this assignment, the result was very bad. Even if I used the weighted model technique, I still got bad results (accuracy scores are very unstable).

In order to overcome this situation, I started to do explorative data analysis and I found that 'fmlwgt' is almost unrelated with 'Target'.

Therefore, I decided to drop 'fmlwgt'. Fortunately, my models started to improve and got trustable predictions.

In short, never forget to do EDA before modeling.