```
In [ ]:
```

The neural network varies form statistical approach. It is more sensitive and prone to overfitting and ignoring the minority classes especially for binary classification model. Since the network is learning the result, the data should be balanced to produce an accurate representation of the model based on fairly proportional input. The readmission binary categorical classification variable of the model is imbalanced. The number of positive readmitted patients is much larger than those whom did not readmitted. The ratio of positive reemission is over 10 times the size of the not readmitted which indicates that more than 90% of the classification variable are of the same class. The ANN sequential model with sigmoid will produce an overfitting result with high accuracy of 91 % and very low precision and recall. In addition the model will not exhibit any learning process. The lost and accuracy will remain constant throughout the learning process. To overcome the problem a different approaches where conducted. The class-weigh and SMOTE methods where selected to predict the readmission classifications. The weight method treat every instance of minority class as multiple instance of majority class based on the ratio between the two classes. It means the loss function will assign a higher value to these minority instances using the ratio between majority and minority classes. Thus, the loss becomes a weighted average, where the weight of each sample is specified by its corresponding class. The model exhibits moderate predicted accuracy value of 82%. The precision of the model is 91% and the recall is 88%.

The Sampling method utilizes the SMOTE algorithm which an oversampling method based on synthetic minority oversampling technique uses k-nearest neighbors to create synthetic examples of the minority class. It injects the SMOTE method at each iteration. The advantage of this approach is that while standard boosting gives equal weights to all misclassified data, SMOTE gives more examples of the minority class at each boosting step. The model produces and highly correlated result with 92 % accuracy. This result is confirmed by the high precision of 87% and 99% of recall.

In [320]:
```
import numpy
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import keras.backend as K
#K.clear_session()
import numpy
import numpy as np
import pandas

import keras
import keras.utils
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation
from keras.optimizers import SGD
from keras.wrappers.scikit_learn import KerasClassifier
from keras.utils import np_utils
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.preprocessing import LabelEncoder
from sklearn.pipeline import Pipeline
import pandas as pd
import seaborn as sns
seed = 123
%matplotlib inline
```

In [321]:
```
# load dataset
df = pd.read_csv('diabetes_data_preprocessed.csv',low_memory=False)
df=df.drop("Unnamed: 0", axis=1)

# convert age back to integer type
df['age'] = df['age'].astype('int64')
print(df.age.value_counts())
# convert age categories to mid-point values
age_dict = {1:5, 2:15, 3:25, 4:35, 5:45, 6:55, 7:65, 8:75, 9:85, 10:95}
df['age'] = df.age.map(age_dict)

df['race'] = df['race'].replace('?', "Unknown")
sum(df['race']=='?')

col=df.columns
GRGs=list(filter(lambda x: x.startswith("D"), col))
GRGs.remove('DRG')

df=df.drop(GRGs,axis=1)
numerics = list(set(list(df._get_numeric_data().columns)) - {'level2_diag1','p
atient_nbr'
                                                             'encounter_id'})
dataframe=df[numerics]
print(dataframe.shape)
dataframe.columns
```

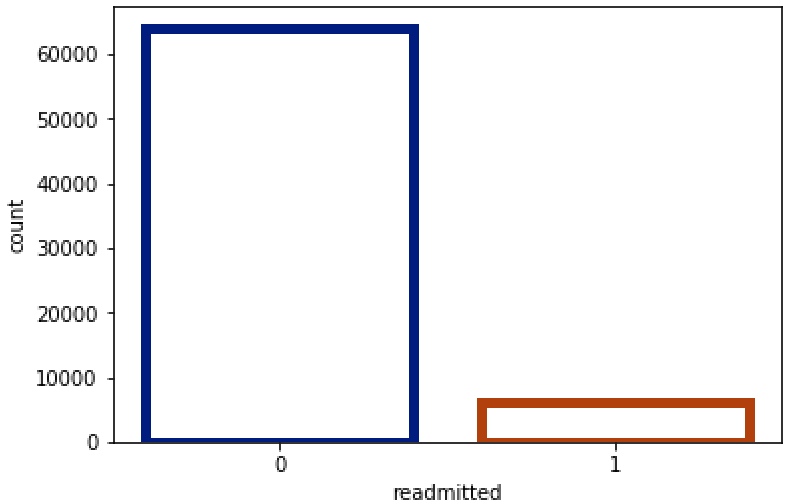
```
8     18098
7     15823
6     12303
9     11533
5      6711
4      2589
10     1898
3      1037
2       360
1        64
Name: age, dtype: int64
(70416, 49)
```

Out[321]:
```
Index(['time_in_hospital', 'level1_diag3', 'level2_diag3',
       'glimepiride.pioglitazone', 'admission_type_id', 'number_outpatient',
       'level2_diag2', 'num_procedures', 'glimepiride',
       'discharge_disposition_id', 'number_inpatient', 'rosiglitazone',
       'metformin.pioglitazone', 'readmitted', 'A1Cresult', 'level1_diag1',
       'glyburide', 'insulin', 'num_medications', 'encounter_id', 'metformi
n',
       'acarbose', 'diabetesMed', 'nummed', 'acetohexamide', 'change',
       'tolbutamide', 'metformin.rosiglitazone', 'DRG', 'troglitazone',
       'max_glu_serum', 'age', 'glyburide.metformin', 'glipizide',
       'number_diagnoses', 'pioglitazone', 'nateglinide', 'level1_diag2',
       'repaglinide', 'glipizide.metformin', 'service_utilization',
       'number_emergency', 'chlorpropamide', 'num_lab_procedures',
       'tolazamide', 'miglitol', 'admission_source_id', 'patient_nbr',
       'numchange'],
      dtype='object')
```

In [ ]:

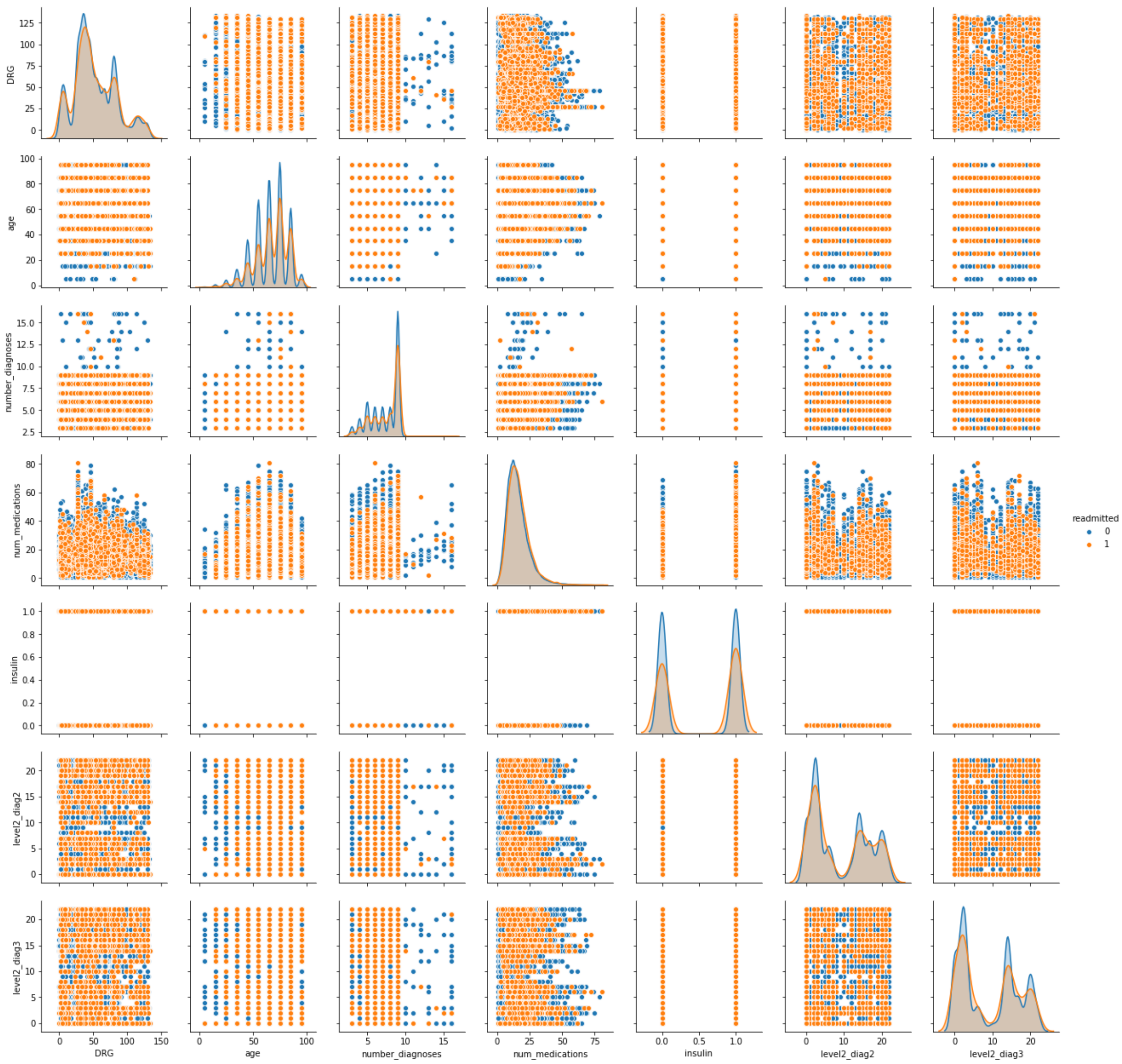
In [322]:
```
sns.countplot(dataframe['readmitted'],facecolor=(0, 0, 0, 0),linewidth=5,edgec
olor=sns.color_palette("dark", 3), label = "Count")
```

Out[322]: <matplotlib.axes._subplots.AxesSubplot at 0x199f6bac8>



In [326]:
```
sns.pairplot(dataframe, hue='readmitted', vars = ['DRG', 'age', 'number_diagnoses', 'num_medications','insulin','level2_diag2','level2_diag3'])
```

Out[326]:
```
<seaborn.axisgrid.PairGrid at 0x1cd968208>
```



In [328]:
```
keys= ['level2_diag2', 'level2_diag3', 'time_in_hospital', 'number_diagnoses',
'DRG',
       'num_medications', 'discharge_disposition_id','admission_source_id',
       'service_utilization', 'age', 'number_outpatient','num_procedures', 'nu
m_lab_procedures', 'admission_type_id']

# Onehotencoder works with a matrix of integers whereas getdummies works with
 a dataframe
def create_dummies(dftemp,column_name):
    dummies = pd.get_dummies(dftemp[column_name],prefix=column_name)
    #print(dummies.columns)
    dftemp = pd.concat([dftemp,dummies],axis=1)
    return dftemp

for i in keys:
    dataframe = create_dummies(dataframe,str(i))
    del dataframe[i]
dataframe.shape
```

Out[328]:
```
(70416, 539)
```

In [329]:
```
y=dataframe.readmitted
print(y[:2])
ratio=sum(y[:]==0)/sum(y[:]==1)
print("ratio :",ratio)
print(y.shape)
```

```
0    0
1    0
Name: readmitted, dtype: int64
ratio : 10.26656
(70416,)
```

In [330]:
```
dataframe=dataframe.drop('readmitted',axis=1)
dataset=dataframe.values
x=dataset.astype(float)
x.shape
print(dataset.shape)
```

```
(70416, 538)
```

In [331]:
```
#standardizing the input feature
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X = sc.fit_transform(x)
X
```

Out[331]:
```
array([[-0.93518286,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827],
       [-0.93518286,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827],
       [-0.48161408,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827],
       ...,
       [-0.48161408,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827],
       [-0.93518286,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827],
       [ 0.42552347,  0.        , -0.23579941, ..., -0.49143731,
        -0.01130611, -0.35593827]])
```

Sequential Model

In [478]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
```

In [479]:
```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
labelencoder_X_1 = LabelEncoder()
from sklearn.utils import class_weight
encoder = LabelEncoder()
encoder.fit(y)
y_train= encoder.transform(y_train)
y_test= encoder.transform(y_test)
class_weight_list =class_weight.compute_class_weight('balanced', numpy.unique(
y_train), y_train)
class_weight = dict(zip(numpy.unique(y_train), class_weight_list))
y_train=keras.utils.np_utils.to_categorical(y_train, 2)
y_test=keras.utils.np_utils.to_categorical(y_test, 2)
```

In [480]:
```
y_test
```

Out[480]:
```
array([[1., 0.],
       [1., 0.],
       [1., 0.],
       ...,
       [1., 0.],
       [1., 0.],
       [1., 0.]], dtype=float32)
```

In [481]:
```
from keras.models import Sequential
from keras.layers import Dense
model = Sequential()
#First Hidden Layer
model.add(Dense(269, activation='relu', kernel_initializer='random_normal', in
put_dim=538))
model.add(Dropout(0.5))
#Second  Hidden Layer
model.add(Dense(269, activation='relu', kernel_initializer='random_normal'))
model.add(Dropout(0.2))
#Second  Hidden Layer
model.add(Dense(134, activation='relu', kernel_initializer='random_normal'))
#Output Layer
model.add(Dense(2, activation='sigmoid', kernel_initializer='random_normal'))
```

In [482]:
```
#Compiling the neural network
model.compile(optimizer ='adam',loss='binary_crossentropy', metrics =['accurac
y'])
```

In [483]:
```
hist = model.fit(X_train,y_train,
               batch_size=20,
               epochs=200,
               class_weight=class_weight,
               validation_data=(X_test, y_test)
              )
```

```
Train on 49291 samples, validate on 21125 samples
Epoch 1/200
49291/49291 [==============================] - 16s 333us/step - loss: 0.6779
- acc: 0.5949 - val_loss: 0.6130 - val_acc: 0.6626
Epoch 2/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.6514
- acc: 0.6400 - val_loss: 0.6492 - val_acc: 0.6098
Epoch 3/200
49291/49291 [==============================] - 12s 240us/step - loss: 0.6384
- acc: 0.6653 - val_loss: 0.7047 - val_acc: 0.6533
Epoch 4/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.6301
- acc: 0.6638 - val_loss: 0.5875 - val_acc: 0.7251
Epoch 5/200
49291/49291 [==============================] - 12s 239us/step - loss: 0.6234
- acc: 0.6792 - val_loss: 0.6340 - val_acc: 0.6806
Epoch 6/200
49291/49291 [==============================] - 12s 247us/step - loss: 0.6101
- acc: 0.6981 - val_loss: 0.5848 - val_acc: 0.7427
Epoch 7/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.5936
- acc: 0.7105 - val_loss: 0.5932 - val_acc: 0.7242
Epoch 8/200
49291/49291 [==============================] - 12s 243us/step - loss: 0.5862
- acc: 0.7282 - val_loss: 0.5380 - val_acc: 0.8167
Epoch 9/200
49291/49291 [==============================] - 12s 238us/step - loss: 0.5719
- acc: 0.7496 - val_loss: 0.6100 - val_acc: 0.7291
Epoch 10/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.5596
- acc: 0.7586 - val_loss: 0.5901 - val_acc: 0.7341
Epoch 11/200
49291/49291 [==============================] - 12s 246us/step - loss: 0.5413
- acc: 0.7800 - val_loss: 0.4976 - val_acc: 0.8432
Epoch 12/200
49291/49291 [==============================] - 12s 251us/step - loss: 0.5403
- acc: 0.7802 - val_loss: 0.4647 - val_acc: 0.8521
Epoch 13/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.5247
- acc: 0.7972 - val_loss: 0.4749 - val_acc: 0.8542
Epoch 14/200
49291/49291 [==============================] - 12s 239us/step - loss: 0.5168
- acc: 0.7986 - val_loss: 0.4644 - val_acc: 0.8439
Epoch 15/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.5033
- acc: 0.8088 - val_loss: 0.4686 - val_acc: 0.8470
Epoch 16/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.4951
- acc: 0.8198 - val_loss: 0.4582 - val_acc: 0.8517
Epoch 17/200
49291/49291 [==============================] - 12s 238us/step - loss: 0.4877
- acc: 0.8211 - val_loss: 0.4378 - val_acc: 0.8672
Epoch 18/200
49291/49291 [==============================] - 12s 240us/step - loss: 0.4855
- acc: 0.8267 - val_loss: 0.5056 - val_acc: 0.8491
Epoch 19/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.4799
```

```
- acc: 0.8322 - val_loss: 0.4420 - val_acc: 0.8500
Epoch 20/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.4689
- acc: 0.8352 - val_loss: 0.4071 - val_acc: 0.8791
Epoch 21/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4607
- acc: 0.8404 - val_loss: 0.4133 - val_acc: 0.8729
Epoch 22/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4502
- acc: 0.8392 - val_loss: 0.4763 - val_acc: 0.8627
Epoch 23/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.4578
- acc: 0.8467 - val_loss: 0.4245 - val_acc: 0.8642
Epoch 24/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.4456
- acc: 0.8490 - val_loss: 0.3817 - val_acc: 0.8904
Epoch 25/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4427
- acc: 0.8548 - val_loss: 0.4126 - val_acc: 0.8744
Epoch 26/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4358
- acc: 0.8519 - val_loss: 0.4411 - val_acc: 0.8483
Epoch 27/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.4276
- acc: 0.8530 - val_loss: 0.3844 - val_acc: 0.8862
Epoch 28/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.4302
- acc: 0.8558 - val_loss: 0.3808 - val_acc: 0.8875
Epoch 29/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4250
- acc: 0.8600 - val_loss: 0.4157 - val_acc: 0.8644
Epoch 30/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.4151
- acc: 0.8629 - val_loss: 0.3998 - val_acc: 0.8743
Epoch 31/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.4114
- acc: 0.8576 - val_loss: 0.3909 - val_acc: 0.8818
Epoch 32/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.4063
- acc: 0.8661 - val_loss: 0.3871 - val_acc: 0.8827
Epoch 33/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.4186
- acc: 0.8627 - val_loss: 0.4056 - val_acc: 0.8705
Epoch 34/200
49291/49291 [==============================] - 12s 241us/step - loss: 0.4049
- acc: 0.8687 - val_loss: 0.3941 - val_acc: 0.8770
Epoch 35/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.4023
- acc: 0.8683 - val_loss: 0.3842 - val_acc: 0.8807
Epoch 36/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.3933
- acc: 0.8691 - val_loss: 0.3898 - val_acc: 0.8794
Epoch 37/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.3983
- acc: 0.8704 - val_loss: 0.3753 - val_acc: 0.8897
Epoch 38/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.3870
```

```
- acc: 0.8716 - val_loss: 0.3767 - val_acc: 0.8871
Epoch 39/200
49291/49291 [==============================] - 12s 247us/step - loss: 0.3879
- acc: 0.8726 - val_loss: 0.3861 - val_acc: 0.8784
Epoch 40/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.3848
- acc: 0.8737 - val_loss: 0.3581 - val_acc: 0.8954
Epoch 41/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.3742
- acc: 0.8756 - val_loss: 0.3828 - val_acc: 0.8886
Epoch 42/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.3807
- acc: 0.8739 - val_loss: 0.3985 - val_acc: 0.8778
Epoch 43/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3760
- acc: 0.8753 - val_loss: 0.3761 - val_acc: 0.8848
Epoch 44/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.3754
- acc: 0.8785 - val_loss: 0.3924 - val_acc: 0.8776
Epoch 45/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.3713
- acc: 0.8835 - val_loss: 0.3941 - val_acc: 0.8788
Epoch 46/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3668
- acc: 0.8825 - val_loss: 0.3710 - val_acc: 0.8872
Epoch 47/200
49291/49291 [==============================] - 12s 238us/step - loss: 0.3815
- acc: 0.8789 - val_loss: 0.3911 - val_acc: 0.8795
Epoch 48/200
49291/49291 [==============================] - 12s 238us/step - loss: 0.3721
- acc: 0.8801 - val_loss: 0.4043 - val_acc: 0.8715
Epoch 49/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3671
- acc: 0.8800 - val_loss: 0.3909 - val_acc: 0.8800
Epoch 50/200
49291/49291 [==============================] - 12s 243us/step - loss: 0.3750
- acc: 0.8826 - val_loss: 0.4046 - val_acc: 0.8755
Epoch 51/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.3598
- acc: 0.8849 - val_loss: 0.3879 - val_acc: 0.8851
Epoch 52/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.3517
- acc: 0.8880 - val_loss: 0.4913 - val_acc: 0.8415
Epoch 53/200
49291/49291 [==============================] - 12s 243us/step - loss: 0.3534
- acc: 0.8837 - val_loss: 0.3783 - val_acc: 0.8889
Epoch 54/200
49291/49291 [==============================] - 16s 327us/step - loss: 0.3467
- acc: 0.8863 - val_loss: 0.3981 - val_acc: 0.8835
Epoch 55/200
49291/49291 [==============================] - 16s 324us/step - loss: 0.3526
- acc: 0.8851 - val_loss: 0.4055 - val_acc: 0.8766
Epoch 56/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3429
- acc: 0.8833 - val_loss: 0.4054 - val_acc: 0.8716
Epoch 57/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3587
```

```
- acc: 0.8822 - val_loss: 0.3929 - val_acc: 0.8794
Epoch 58/200
49291/49291 [==============================] - 16s 316us/step - loss: 0.3509
- acc: 0.8857 - val_loss: 0.4118 - val_acc: 0.8669
Epoch 59/200
49291/49291 [==============================] - 15s 311us/step - loss: 0.3590
- acc: 0.8870 - val_loss: 0.4110 - val_acc: 0.8753
Epoch 60/200
49291/49291 [==============================] - 15s 311us/step - loss: 0.3555
- acc: 0.8878 - val_loss: 0.4073 - val_acc: 0.8844
Epoch 61/200
49291/49291 [==============================] - 15s 313us/step - loss: 0.3372
- acc: 0.8878 - val_loss: 0.4191 - val_acc: 0.8572
Epoch 62/200
49291/49291 [==============================] - 15s 314us/step - loss: 0.3399
- acc: 0.8917 - val_loss: 0.4099 - val_acc: 0.8818
Epoch 63/200
49291/49291 [==============================] - 16s 328us/step - loss: 0.3445
- acc: 0.8904 - val_loss: 0.3950 - val_acc: 0.8762
Epoch 64/200
49291/49291 [==============================] - 16s 324us/step - loss: 0.3389
- acc: 0.8875 - val_loss: 0.4171 - val_acc: 0.8792
Epoch 65/200
49291/49291 [==============================] - 16s 316us/step - loss: 0.3360
- acc: 0.8874 - val_loss: 0.3957 - val_acc: 0.8876
Epoch 66/200
49291/49291 [==============================] - 16s 318us/step - loss: 0.3424
- acc: 0.8849 - val_loss: 0.4035 - val_acc: 0.8738
Epoch 67/200
49291/49291 [==============================] - 16s 315us/step - loss: 0.3295
- acc: 0.8894 - val_loss: 0.3968 - val_acc: 0.8800
Epoch 68/200
49291/49291 [==============================] - 16s 315us/step - loss: 0.3557
- acc: 0.8839 - val_loss: 0.3941 - val_acc: 0.8809
Epoch 69/200
49291/49291 [==============================] - 15s 309us/step - loss: 0.3388
- acc: 0.8871 - val_loss: 0.4309 - val_acc: 0.8806
Epoch 70/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3259
- acc: 0.8909 - val_loss: 0.3965 - val_acc: 0.8778
Epoch 71/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3306
- acc: 0.8933 - val_loss: 0.4214 - val_acc: 0.8654
Epoch 72/200
49291/49291 [==============================] - 16s 322us/step - loss: 0.3347
- acc: 0.8931 - val_loss: 0.4006 - val_acc: 0.8754
Epoch 73/200
49291/49291 [==============================] - 16s 330us/step - loss: 0.3210
- acc: 0.8939 - val_loss: 0.4060 - val_acc: 0.8721
Epoch 74/200
49291/49291 [==============================] - 16s 318us/step - loss: 0.3281
- acc: 0.8907 - val_loss: 0.3947 - val_acc: 0.8755
Epoch 75/200
49291/49291 [==============================] - 16s 326us/step - loss: 0.3436
- acc: 0.8893 - val_loss: 0.4018 - val_acc: 0.8800
Epoch 76/200
49291/49291 [==============================] - 16s 319us/step - loss: 0.3285
```

```
- acc: 0.8922 - val_loss: 0.3889 - val_acc: 0.8845
Epoch 77/200
49291/49291 [==============================] - 16s 315us/step - loss: 0.3211
- acc: 0.8896 - val_loss: 0.4283 - val_acc: 0.8612
Epoch 78/200
49291/49291 [==============================] - 15s 314us/step - loss: 0.3271
- acc: 0.8925 - val_loss: 0.3998 - val_acc: 0.8753
Epoch 79/200
49291/49291 [==============================] - 16s 318us/step - loss: 0.3242
- acc: 0.8897 - val_loss: 0.4008 - val_acc: 0.8804
Epoch 80/200
49291/49291 [==============================] - 16s 318us/step - loss: 0.3339
- acc: 0.8871 - val_loss: 0.4077 - val_acc: 0.8704
Epoch 81/200
49291/49291 [==============================] - 16s 316us/step - loss: 0.3449
- acc: 0.8890 - val_loss: 0.4090 - val_acc: 0.8799
Epoch 82/200
49291/49291 [==============================] - 16s 321us/step - loss: 0.3286
- acc: 0.8957 - val_loss: 0.4028 - val_acc: 0.8747
Epoch 83/200
49291/49291 [==============================] - 16s 321us/step - loss: 0.3267
- acc: 0.8897 - val_loss: 0.4109 - val_acc: 0.8664
Epoch 84/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3314
- acc: 0.8924 - val_loss: 0.3949 - val_acc: 0.8765
Epoch 85/200
49291/49291 [==============================] - 15s 313us/step - loss: 0.3225
- acc: 0.8868 - val_loss: 0.4045 - val_acc: 0.8696
Epoch 86/200
49291/49291 [==============================] - 16s 318us/step - loss: 0.3386
- acc: 0.8927 - val_loss: 0.4304 - val_acc: 0.8564
Epoch 87/200
49291/49291 [==============================] - 15s 312us/step - loss: 0.3287
- acc: 0.8930 - val_loss: 0.4043 - val_acc: 0.8688
Epoch 88/200
49291/49291 [==============================] - 16s 316us/step - loss: 0.3261
- acc: 0.8946 - val_loss: 0.3983 - val_acc: 0.8779
Epoch 89/200
49291/49291 [==============================] - 15s 309us/step - loss: 0.3212
- acc: 0.8947 - val_loss: 0.4143 - val_acc: 0.8789
Epoch 90/200
49291/49291 [==============================] - 16s 316us/step - loss: 0.3134
- acc: 0.8928 - val_loss: 0.3987 - val_acc: 0.8733
Epoch 91/200
49291/49291 [==============================] - 15s 307us/step - loss: 0.3227
- acc: 0.8954 - val_loss: 0.4193 - val_acc: 0.8668
Epoch 92/200
49291/49291 [==============================] - 16s 332us/step - loss: 0.3208
- acc: 0.8973 - val_loss: 0.4127 - val_acc: 0.8701
Epoch 93/200
49291/49291 [==============================] - 16s 325us/step - loss: 0.3170
- acc: 0.8971 - val_loss: 0.4178 - val_acc: 0.8796
Epoch 94/200
49291/49291 [==============================] - 16s 333us/step - loss: 0.3215
- acc: 0.8960 - val_loss: 0.4138 - val_acc: 0.8739
Epoch 95/200
49291/49291 [==============================] - 16s 327us/step - loss: 0.3218
```

```
- acc: 0.8956 - val_loss: 0.4171 - val_acc: 0.8664
Epoch 96/200
49291/49291 [==============================] - 17s 337us/step - loss: 0.3272
- acc: 0.8939 - val_loss: 0.3984 - val_acc: 0.8802
Epoch 97/200
49291/49291 [==============================] - 16s 332us/step - loss: 0.3476
- acc: 0.8953 - val_loss: 0.4171 - val_acc: 0.8671
Epoch 98/200
49291/49291 [==============================] - 16s 329us/step - loss: 0.3320
- acc: 0.8966 - val_loss: 0.4282 - val_acc: 0.8691
Epoch 99/200
49291/49291 [==============================] - 15s 297us/step - loss: 0.3166
- acc: 0.8954 - val_loss: 0.4231 - val_acc: 0.8649
Epoch 100/200
49291/49291 [==============================] - 13s 256us/step - loss: 0.3187
- acc: 0.8971 - val_loss: 0.4064 - val_acc: 0.8712
Epoch 101/200
49291/49291 [==============================] - 12s 245us/step - loss: 0.3316
- acc: 0.8928 - val_loss: 0.4250 - val_acc: 0.8604
Epoch 102/200
49291/49291 [==============================] - 12s 253us/step - loss: 0.3140
- acc: 0.8958 - val_loss: 0.4177 - val_acc: 0.8664
Epoch 103/200
49291/49291 [==============================] - 12s 247us/step - loss: 0.3155
- acc: 0.8984 - val_loss: 0.4052 - val_acc: 0.8705
Epoch 104/200
49291/49291 [==============================] - 12s 253us/step - loss: 0.3150
- acc: 0.9003 - val_loss: 0.4075 - val_acc: 0.8769
Epoch 105/200
49291/49291 [==============================] - 12s 249us/step - loss: 0.3186
- acc: 0.8953 - val_loss: 0.4025 - val_acc: 0.8817
Epoch 106/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.3197
- acc: 0.8982 - val_loss: 0.4143 - val_acc: 0.8745
Epoch 107/200
49291/49291 [==============================] - 12s 251us/step - loss: 0.3128
- acc: 0.8965 - val_loss: 0.4159 - val_acc: 0.8780
Epoch 108/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.3157
- acc: 0.8962 - val_loss: 0.3991 - val_acc: 0.8819
Epoch 109/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.2969
- acc: 0.9033 - val_loss: 0.4007 - val_acc: 0.8800
Epoch 110/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.3282
- acc: 0.8949 - val_loss: 0.4278 - val_acc: 0.8640
Epoch 111/200
49291/49291 [==============================] - 12s 245us/step - loss: 0.3253
- acc: 0.8946 - val_loss: 0.4089 - val_acc: 0.8747
Epoch 112/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.3027
- acc: 0.9012 - val_loss: 0.4108 - val_acc: 0.8736
Epoch 113/200
49291/49291 [==============================] - 12s 247us/step - loss: 0.3239
- acc: 0.8991 - val_loss: 0.4371 - val_acc: 0.8573
Epoch 114/200
49291/49291 [==============================] - 13s 256us/step - loss: 0.3063
```

```
- acc: 0.9044 - val_loss: 0.4071 - val_acc: 0.8674
Epoch 115/200
49291/49291 [==============================] - 13s 255us/step - loss: 0.3122
- acc: 0.9000 - val_loss: 0.4005 - val_acc: 0.8779
Epoch 116/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.3123
- acc: 0.9000 - val_loss: 0.4219 - val_acc: 0.8621
Epoch 117/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.3159
- acc: 0.8972 - val_loss: 0.4122 - val_acc: 0.8703
Epoch 118/200
49291/49291 [==============================] - 13s 256us/step - loss: 0.3101
- acc: 0.9010 - val_loss: 0.4325 - val_acc: 0.8534
Epoch 119/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.3185
- acc: 0.8998 - val_loss: 0.4056 - val_acc: 0.8817
Epoch 120/200
49291/49291 [==============================] - 12s 253us/step - loss: 0.3047
- acc: 0.9012 - val_loss: 0.3988 - val_acc: 0.8808
Epoch 121/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.3070
- acc: 0.9015 - val_loss: 0.4156 - val_acc: 0.8605
Epoch 122/200
49291/49291 [==============================] - 12s 248us/step - loss: 0.3012
- acc: 0.8991 - val_loss: 0.4137 - val_acc: 0.8684
Epoch 123/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.3089
- acc: 0.9001 - val_loss: 0.3922 - val_acc: 0.8831
Epoch 124/200
49291/49291 [==============================] - 12s 251us/step - loss: 0.3101
- acc: 0.8977 - val_loss: 0.4097 - val_acc: 0.8779
Epoch 125/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.3183
- acc: 0.8943 - val_loss: 0.4145 - val_acc: 0.8646
Epoch 126/200
49291/49291 [==============================] - 13s 257us/step - loss: 0.2989
- acc: 0.9023 - val_loss: 0.4088 - val_acc: 0.8745
Epoch 127/200
49291/49291 [==============================] - 12s 253us/step - loss: 0.3125
- acc: 0.8980 - val_loss: 0.4295 - val_acc: 0.8607
Epoch 128/200
49291/49291 [==============================] - 13s 255us/step - loss: 0.3099
- acc: 0.9004 - val_loss: 0.4224 - val_acc: 0.8733
Epoch 129/200
49291/49291 [==============================] - 13s 254us/step - loss: 0.3205
- acc: 0.8963 - val_loss: 0.4114 - val_acc: 0.8728
Epoch 130/200
49291/49291 [==============================] - 13s 259us/step - loss: 0.3262
- acc: 0.8971 - val_loss: 0.4217 - val_acc: 0.8637
Epoch 131/200
49291/49291 [==============================] - 12s 247us/step - loss: 0.3008
- acc: 0.9023 - val_loss: 0.4114 - val_acc: 0.8659
Epoch 132/200
49291/49291 [==============================] - 13s 258us/step - loss: 0.3182
- acc: 0.8972 - val_loss: 0.4046 - val_acc: 0.8687
Epoch 133/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.3110
```

```
- acc: 0.9007 - val_loss: 0.4109 - val_acc: 0.8740
Epoch 134/200
49291/49291 [==============================] - 12s 253us/step - loss: 0.3135
- acc: 0.9003 - val_loss: 0.4010 - val_acc: 0.8733
Epoch 135/200
49291/49291 [==============================] - 12s 246us/step - loss: 0.2965
- acc: 0.9070 - val_loss: 0.3958 - val_acc: 0.8802
Epoch 136/200
49291/49291 [==============================] - 13s 254us/step - loss: 0.3030
- acc: 0.9026 - val_loss: 0.4280 - val_acc: 0.8543
Epoch 137/200
49291/49291 [==============================] - 12s 250us/step - loss: 0.3057
- acc: 0.9053 - val_loss: 0.4034 - val_acc: 0.8708
Epoch 138/200
49291/49291 [==============================] - 13s 254us/step - loss: 0.2996
- acc: 0.9023 - val_loss: 0.4114 - val_acc: 0.8670
Epoch 139/200
49291/49291 [==============================] - 12s 252us/step - loss: 0.2999
- acc: 0.9002 - val_loss: 0.4047 - val_acc: 0.8739
Epoch 140/200
49291/49291 [==============================] - 13s 258us/step - loss: 0.3065
- acc: 0.9030 - val_loss: 0.4219 - val_acc: 0.8702
Epoch 141/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.3044
- acc: 0.9008 - val_loss: 0.4015 - val_acc: 0.8786
Epoch 142/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.2983
- acc: 0.9038 - val_loss: 0.4186 - val_acc: 0.8631
Epoch 143/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3066
- acc: 0.9006 - val_loss: 0.4021 - val_acc: 0.8743
Epoch 144/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3097
- acc: 0.8991 - val_loss: 0.4138 - val_acc: 0.8666
Epoch 145/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3063
- acc: 0.9025 - val_loss: 0.4053 - val_acc: 0.8687
Epoch 146/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.3050
- acc: 0.9039 - val_loss: 0.4104 - val_acc: 0.8699
Epoch 147/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3066
- acc: 0.8989 - val_loss: 0.4002 - val_acc: 0.8734
Epoch 148/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.2986
- acc: 0.9026 - val_loss: 0.4061 - val_acc: 0.8748
Epoch 149/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.3085
- acc: 0.9012 - val_loss: 0.4036 - val_acc: 0.8747
Epoch 150/200
49291/49291 [==============================] - 12s 243us/step - loss: 0.2982
- acc: 0.9007 - val_loss: 0.4172 - val_acc: 0.8685
Epoch 151/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.3056
- acc: 0.9024 - val_loss: 0.4110 - val_acc: 0.8710
Epoch 152/200
49291/49291 [==============================] - 12s 242us/step - loss: 0.3118
```

```
- acc: 0.9053 - val_loss: 0.4066 - val_acc: 0.8744
Epoch 153/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.2987
- acc: 0.9043 - val_loss: 0.4056 - val_acc: 0.8768
Epoch 154/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3070
- acc: 0.9047 - val_loss: 0.4194 - val_acc: 0.8636
Epoch 155/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3174
- acc: 0.9037 - val_loss: 0.4180 - val_acc: 0.8677
Epoch 156/200
49291/49291 [==============================] - 12s 241us/step - loss: 0.3086
- acc: 0.9003 - val_loss: 0.4097 - val_acc: 0.8720
Epoch 157/200
49291/49291 [==============================] - 12s 245us/step - loss: 0.3085
- acc: 0.9006 - val_loss: 0.4154 - val_acc: 0.8694
Epoch 158/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3097
- acc: 0.9043 - val_loss: 0.4266 - val_acc: 0.8669
Epoch 159/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.3067
- acc: 0.9097 - val_loss: 0.4226 - val_acc: 0.8675
Epoch 160/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.3142
- acc: 0.9046 - val_loss: 0.4124 - val_acc: 0.8789
Epoch 161/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3189
- acc: 0.9035 - val_loss: 0.4239 - val_acc: 0.8700
Epoch 162/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.2990
- acc: 0.9055 - val_loss: 0.4058 - val_acc: 0.8783
Epoch 163/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.3108
- acc: 0.9030 - val_loss: 0.4132 - val_acc: 0.8653
Epoch 164/200
49291/49291 [==============================] - 12s 244us/step - loss: 0.3116
- acc: 0.9013 - val_loss: 0.4463 - val_acc: 0.8539
Epoch 165/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.3086
- acc: 0.8992 - val_loss: 0.3983 - val_acc: 0.8849
Epoch 166/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.3134
- acc: 0.9033 - val_loss: 0.4095 - val_acc: 0.8782
Epoch 167/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.3056
- acc: 0.9070 - val_loss: 0.4204 - val_acc: 0.8630
Epoch 168/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.2947
- acc: 0.9064 - val_loss: 0.4235 - val_acc: 0.8647
Epoch 169/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.3081
- acc: 0.9060 - val_loss: 0.4359 - val_acc: 0.8630
Epoch 170/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3214
- acc: 0.9009 - val_loss: 0.4241 - val_acc: 0.8584
Epoch 171/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.2998
```

```
- acc: 0.9045 - val_loss: 0.4150 - val_acc: 0.8703
Epoch 172/200
49291/49291 [==============================] - 12s 233us/step - loss: 0.3001
- acc: 0.9060 - val_loss: 0.4168 - val_acc: 0.8709
Epoch 173/200
49291/49291 [==============================] - 11s 231us/step - loss: 0.3114
- acc: 0.9013 - val_loss: 0.4198 - val_acc: 0.8604
Epoch 174/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3031
- acc: 0.9045 - val_loss: 0.4559 - val_acc: 0.8432
Epoch 175/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3113
- acc: 0.9055 - val_loss: 0.4429 - val_acc: 0.8550
Epoch 176/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3111
- acc: 0.9055 - val_loss: 0.4402 - val_acc: 0.8528
Epoch 177/200
49291/49291 [==============================] - 11s 225us/step - loss: 0.3091
- acc: 0.9012 - val_loss: 0.4306 - val_acc: 0.8623
Epoch 178/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.3085
- acc: 0.9049 - val_loss: 0.4886 - val_acc: 0.8351
Epoch 179/200
49291/49291 [==============================] - 11s 233us/step - loss: 0.3156
- acc: 0.8991 - val_loss: 0.4216 - val_acc: 0.8686
Epoch 180/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3144
- acc: 0.9044 - val_loss: 0.4237 - val_acc: 0.8719
Epoch 181/200
49291/49291 [==============================] - 11s 226us/step - loss: 0.3027
- acc: 0.9091 - val_loss: 0.4370 - val_acc: 0.8629
Epoch 182/200
49291/49291 [==============================] - 11s 227us/step - loss: 0.3146
- acc: 0.9035 - val_loss: 0.4183 - val_acc: 0.8635
Epoch 183/200
49291/49291 [==============================] - 11s 232us/step - loss: 0.3066
- acc: 0.9034 - val_loss: 0.4252 - val_acc: 0.8637
Epoch 184/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.3136
- acc: 0.9018 - val_loss: 0.4121 - val_acc: 0.8809
Epoch 185/200
49291/49291 [==============================] - 12s 244us/step - loss: 0.3077
- acc: 0.9037 - val_loss: 0.4445 - val_acc: 0.8504
Epoch 186/200
49291/49291 [==============================] - 12s 236us/step - loss: 0.3089
- acc: 0.9028 - val_loss: 0.4225 - val_acc: 0.8631
Epoch 187/200
49291/49291 [==============================] - 11s 227us/step - loss: 0.3236
- acc: 0.9010 - val_loss: 0.4240 - val_acc: 0.8697
Epoch 188/200
49291/49291 [==============================] - 12s 235us/step - loss: 0.3061
- acc: 0.9063 - val_loss: 0.4461 - val_acc: 0.8629
Epoch 189/200
49291/49291 [==============================] - 11s 225us/step - loss: 0.3053
- acc: 0.9048 - val_loss: 0.4325 - val_acc: 0.8657
Epoch 190/200
49291/49291 [==============================] - 12s 237us/step - loss: 0.3110
```

```
- acc: 0.9056 - val_loss: 0.4124 - val_acc: 0.8689
Epoch 191/200
49291/49291 [==============================] - 12s 239us/step - loss: 0.3026
- acc: 0.9082 - val_loss: 0.4212 - val_acc: 0.8698
Epoch 192/200
49291/49291 [==============================] - 12s 238us/step - loss: 0.2964
- acc: 0.9087 - val_loss: 0.4303 - val_acc: 0.8641
Epoch 193/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3090
- acc: 0.9071 - val_loss: 0.4279 - val_acc: 0.8656
Epoch 194/200
49291/49291 [==============================] - 11s 228us/step - loss: 0.3142
- acc: 0.9029 - val_loss: 0.4134 - val_acc: 0.8742
Epoch 195/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.3074
- acc: 0.9042 - val_loss: 0.4152 - val_acc: 0.8673
Epoch 196/200
49291/49291 [==============================] - 12s 234us/step - loss: 0.3095
- acc: 0.9038 - val_loss: 0.4395 - val_acc: 0.8587
Epoch 197/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3068
- acc: 0.9028 - val_loss: 0.4188 - val_acc: 0.8673
Epoch 198/200
49291/49291 [==============================] - 13s 255us/step - loss: 0.3054
- acc: 0.9056 - val_loss: 0.4258 - val_acc: 0.8683
Epoch 199/200
49291/49291 [==============================] - 11s 230us/step - loss: 0.3065
- acc: 0.9062 - val_loss: 0.4255 - val_acc: 0.8674
Epoch 200/200
49291/49291 [==============================] - 11s 229us/step - loss: 0.2946
- acc: 0.9074 - val_loss: 0.4853 - val_acc: 0.8282
```

In [484]:

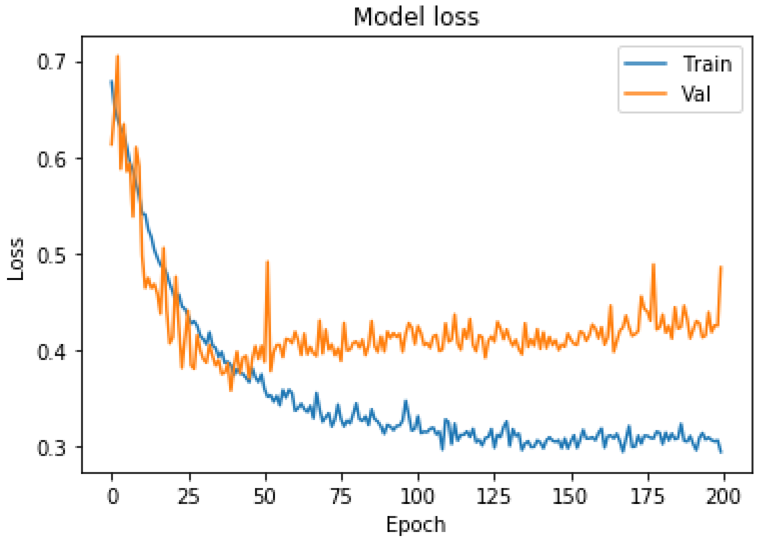
```
model.evaluate(X_test, y_test)[1]
```

```
21125/21125 [==============================] - 1s 47us/step
```

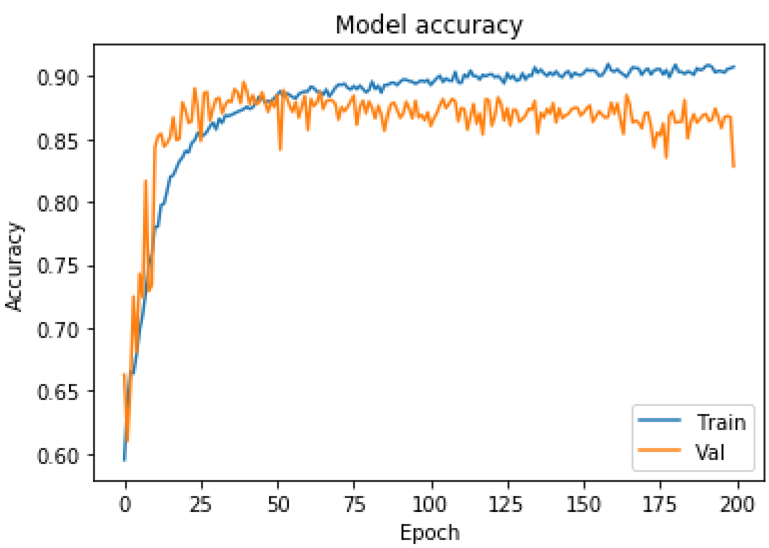
Out[484]:

```
0.8282366863848895
```

In [485]:
```
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.show()
```



In [486]:
```
plt.plot(hist.history['acc'])
plt.plot(hist.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='lower right')
plt.show()
```



In [525]:
```
y_test_pred = (model.predict(X_test)>.5)
from sklearn.metrics import accuracy_score, precision_score, recall_score
Recall=recall_score(y_test,y_test_pred, average=None)
Precision=precision_score(y_test,y_test_pred, average=None)
ACC=accuracy_score(y_test,y_test_pred)
print("acu: ",round(ACC,4))
print("precision:",Precision[0])
print("recall:",Recall[0])
```

```
acu:  0.8228
precision: 0.9169440566949425
recall: 0.8868061685445766
```

SMOTE Model

In [341]:
```
X_resample,y_resample=SMOTE().fit_sample(X,y.values.ravel())
```

In [342]:
```
y_resample=pd.DataFrame(y_resample)
X_resample=pd.DataFrame(X_resample)
```

In [343]:
```
X_train, X_test, y_train, y_test = train_test_split(X_resample,
                                                     y_resample, test_size = 0.
3,
                                                     random_state=0)
```

In [344]:
```
X_train = np.array(X_train)
X_test=np.array(X_test)
y_train=np.array(y_train)
y_test=np.array(y_test)
```

In [364]:
```
from keras.models import Sequential
from keras.layers import Dense
model = Sequential()
#First Hidden Layer
model.add(Dense(269, activation='relu', kernel_initializer='random_normal', in
put_dim=538))
model.add(Dropout(0.2))
#Second  Hidden Layer
model.add(Dense(134, activation='relu', kernel_initializer='random_normal'))
model.add(Dropout(0.2))
#Second  Hidden Layer
model.add(Dense(75, activation='relu', kernel_initializer='random_normal'))
#Output Layer
model.add(Dense(1, activation='sigmoid', kernel_initializer='random_normal'))
```

In [365]:
```
model.compile(optimizer='adam', loss='binary_crossentropy',metrics=['accuracy'
])
hist = model.fit(X_train,y_train, batch_size=15, epochs=30,class_weight=class_
weight, validation_data=(X_test, y_test))
```

```
Train on 89832 samples, validate on 38500 samples
Epoch 1/30
89832/89832 [==============================] - 22s 249us/step - loss: 0.7916
- acc: 0.5342 - val_loss: 0.6812 - val_acc: 0.5894
Epoch 2/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.5659
- acc: 0.6676 - val_loss: 0.5703 - val_acc: 0.7234
Epoch 3/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.4559
- acc: 0.7610 - val_loss: 0.4506 - val_acc: 0.8054
Epoch 4/30
89832/89832 [==============================] - 21s 230us/step - loss: 0.3959
- acc: 0.8023 - val_loss: 0.4047 - val_acc: 0.8304
Epoch 5/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.3552
- acc: 0.8298 - val_loss: 0.3459 - val_acc: 0.8541
Epoch 6/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.3294
- acc: 0.8442 - val_loss: 0.3469 - val_acc: 0.8551
Epoch 7/30
89832/89832 [==============================] - 21s 231us/step - loss: 0.3038
- acc: 0.8614 - val_loss: 0.3621 - val_acc: 0.8635
Epoch 8/30
89832/89832 [==============================] - 21s 231us/step - loss: 0.2865
- acc: 0.8744 - val_loss: 0.2800 - val_acc: 0.8842
Epoch 9/30
89832/89832 [==============================] - 21s 232us/step - loss: 0.2705
- acc: 0.8821 - val_loss: 0.3165 - val_acc: 0.8752
Epoch 10/30
89832/89832 [==============================] - 21s 234us/step - loss: 0.2570
- acc: 0.8896 - val_loss: 0.2859 - val_acc: 0.8960
Epoch 11/30
89832/89832 [==============================] - 21s 239us/step - loss: 0.2404
- acc: 0.8972 - val_loss: 0.3176 - val_acc: 0.8785
Epoch 12/30
89832/89832 [==============================] - 21s 233us/step - loss: 0.2367
- acc: 0.9021 - val_loss: 0.2626 - val_acc: 0.9065
Epoch 13/30
89832/89832 [==============================] - 21s 235us/step - loss: 0.2262
- acc: 0.9048 - val_loss: 0.3096 - val_acc: 0.8826
Epoch 14/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.2308
- acc: 0.9061 - val_loss: 0.2761 - val_acc: 0.9077
Epoch 15/30
89832/89832 [==============================] - 21s 230us/step - loss: 0.2156
- acc: 0.9129 - val_loss: 0.2440 - val_acc: 0.9076
Epoch 16/30
89832/89832 [==============================] - 21s 231us/step - loss: 0.2112
- acc: 0.9126 - val_loss: 0.2275 - val_acc: 0.9181
Epoch 17/30
89832/89832 [==============================] - 20s 228us/step - loss: 0.2159
- acc: 0.9161 - val_loss: 0.2436 - val_acc: 0.9102
Epoch 18/30
89832/89832 [==============================] - 20s 227us/step - loss: 0.1996
- acc: 0.9204 - val_loss: 0.2271 - val_acc: 0.9209
Epoch 19/30
89832/89832 [==============================] - 20s 228us/step - loss: 0.2029
```

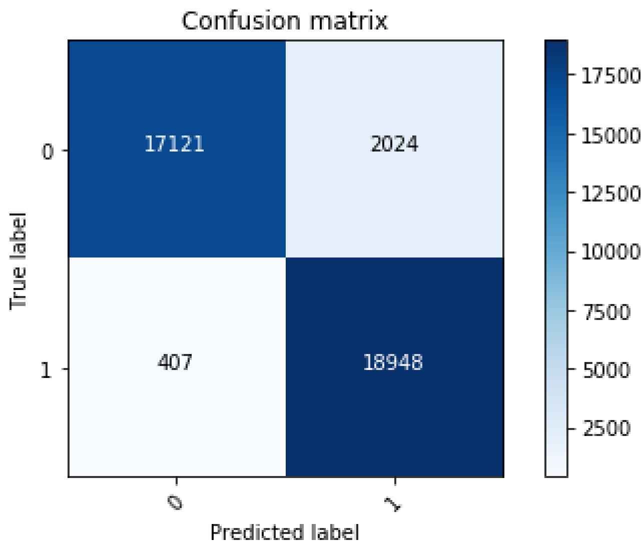
```
- acc: 0.9231 - val_loss: 0.2468 - val_acc: 0.9159
Epoch 20/30
89832/89832 [==============================] - 20s 226us/step - loss: 0.2000
- acc: 0.9224 - val_loss: 0.2556 - val_acc: 0.9150
Epoch 21/30
89832/89832 [==============================] - 21s 229us/step - loss: 0.1953
- acc: 0.9268 - val_loss: 0.2345 - val_acc: 0.9197
Epoch 22/30
89832/89832 [==============================] - 20s 228us/step - loss: 0.1894
- acc: 0.9295 - val_loss: 0.2227 - val_acc: 0.9277
Epoch 23/30
89832/89832 [==============================] - 20s 225us/step - loss: 0.1910
- acc: 0.9301 - val_loss: 0.2137 - val_acc: 0.9293
Epoch 24/30
89832/89832 [==============================] - 20s 227us/step - loss: 0.1888
- acc: 0.9310 - val_loss: 0.1989 - val_acc: 0.9328
Epoch 25/30
89832/89832 [==============================] - 20s 228us/step - loss: 0.1839
- acc: 0.9347 - val_loss: 0.2086 - val_acc: 0.9342
Epoch 26/30
89832/89832 [==============================] - 20s 228us/step - loss: 0.1829
- acc: 0.9361 - val_loss: 0.1937 - val_acc: 0.9370
Epoch 27/30
89832/89832 [==============================] - 20s 226us/step - loss: 0.1833
- acc: 0.9358 - val_loss: 0.2192 - val_acc: 0.9346
Epoch 28/30
89832/89832 [==============================] - 20s 226us/step - loss: 0.1808
- acc: 0.9379 - val_loss: 0.2024 - val_acc: 0.9309
Epoch 29/30
89832/89832 [==============================] - 20s 227us/step - loss: 0.1769
- acc: 0.9375 - val_loss: 0.1793 - val_acc: 0.9425
Epoch 30/30
89832/89832 [==============================] - 20s 227us/step - loss: 0.1772
- acc: 0.9387 - val_loss: 0.2026 - val_acc: 0.9369
```

In [353]:

```
import itertools
def plot_confusion_matrix(cm, classes,
                          normalize=False,
                          title='Confusion matrix',
                          cmap=plt.cm.Blues):
    """
    This function prints and plots the confusion matrix.
    Normalization can be applied by setting `normalize=True`.
    """
    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalized confusion matrix")
    else:
        print('Confusion matrix, without normalization')

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=45)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.
    for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
        plt.text(j, i, format(cm[i, j], fmt),
                 horizontalalignment="center",
                 color="white" if cm[i, j] > thresh else "black")

    plt.ylabel('True label')
    plt.xlabel('Predicted label')
    plt.tight_layout()
```

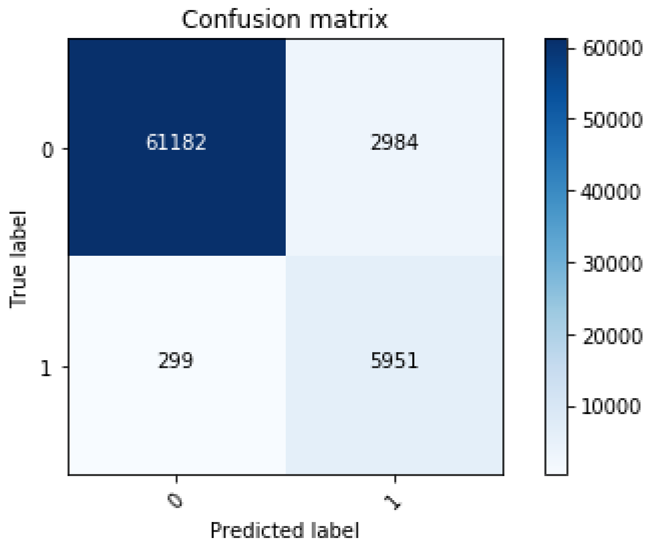
In [366]:
```
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix

y_pred=model.predict(X_test)
y_expected=pd.DataFrame(y_test)

cnf_matrix=confusion_matrix(y_expected,y_pred.round())
plot_confusion_matrix(cnf_matrix,classes=[0,1])
plt.show()
```

```
Confusion matrix, without normalization
[[17121  2024]
 [  407 18948]]
```

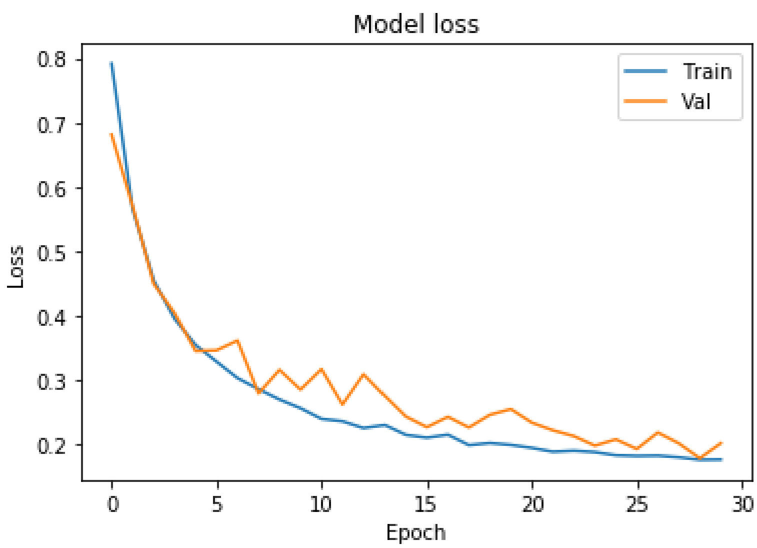


In [367]:
```
y_pred=model.predict(X)

cnf_matrix=confusion_matrix(y,y_pred.round())
plot_confusion_matrix(cnf_matrix,classes=[0,1])
plt.show()
```

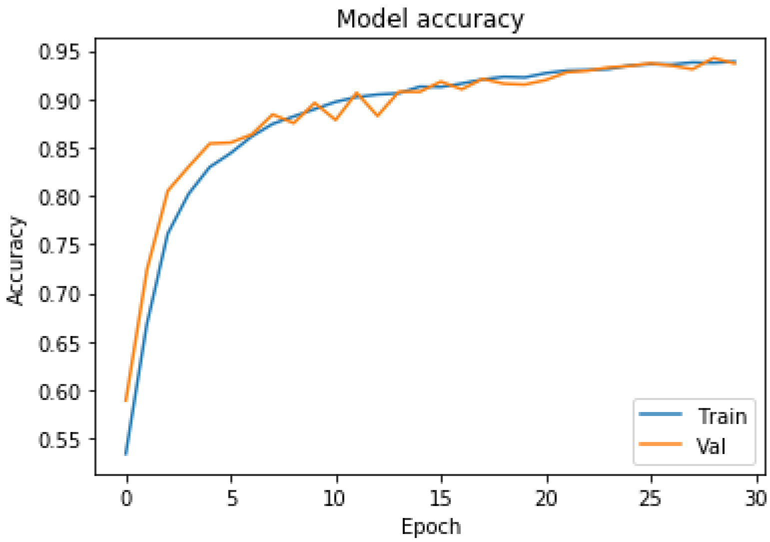
```
Confusion matrix, without normalization
[[61182  2984]
 [  299  5951]]
```



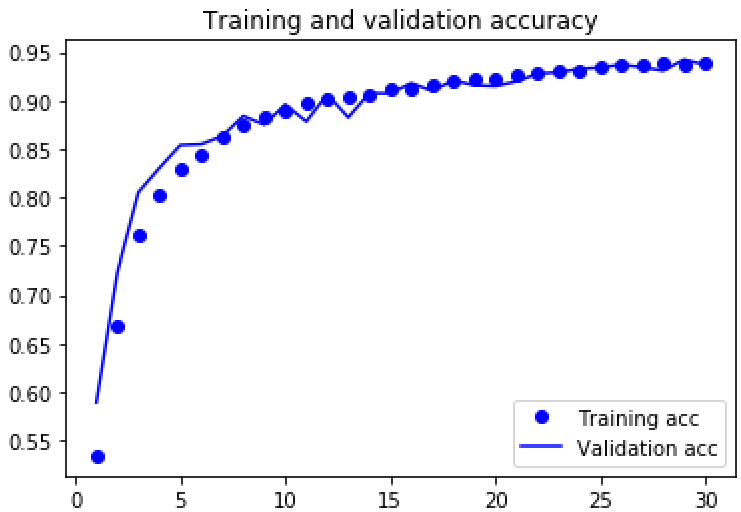
In [368]:
```
plt.plot(hist.history['loss'])
plt.plot(hist.history['val_loss'])
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper right')
plt.show()
```

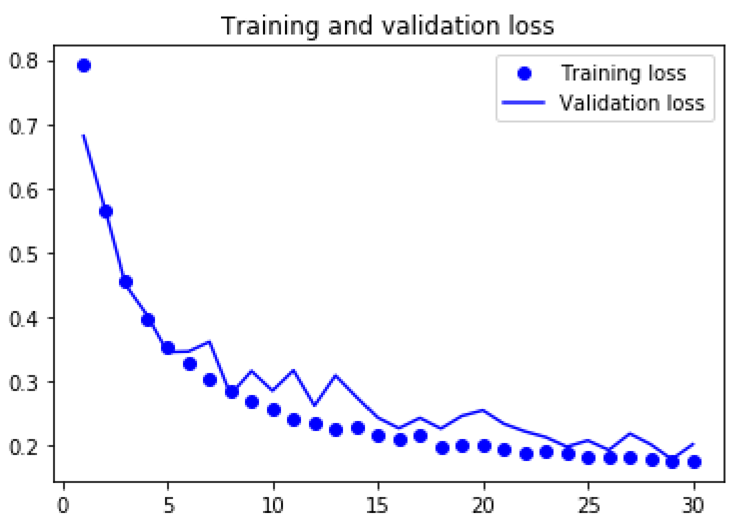


In [369]:
```
plt.plot(hist.history['acc'])
plt.plot(hist.history['val_acc'])
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='lower right')
plt.show()
```



In [376]:
```
acc = hist.history['acc']
val_acc = hist.history['val_acc']
loss = hist.history['loss']
val_loss = hist.history['val_loss']
epochs = range(1, len(acc) + 1)
plt.plot(epochs, acc, 'bo', label='Training acc')
plt.plot(epochs, val_acc, 'b', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.figure()
plt.plot(epochs, loss, 'bo', label='Training loss')
plt.plot(epochs, val_loss, 'b', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.show()
```

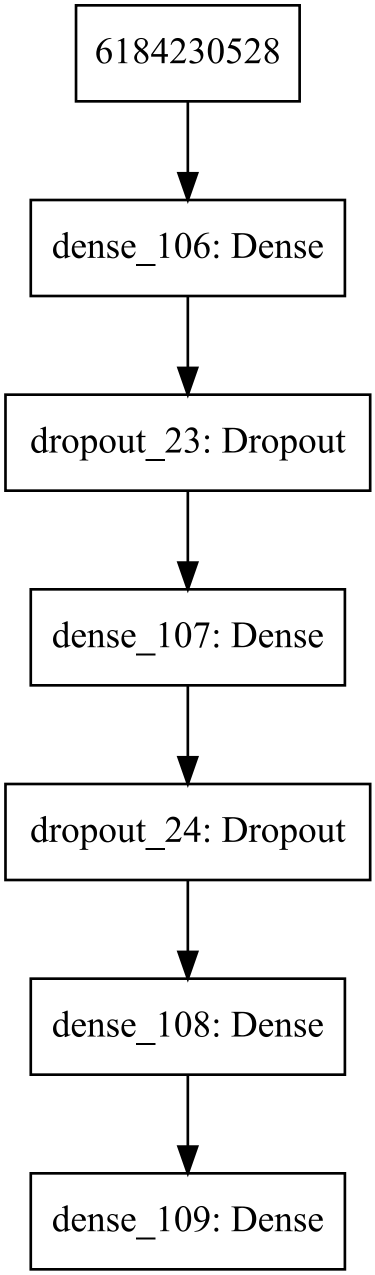




In [370]:

```
from IPython.display import SVG
from keras.utils.vis_utils import model_to_dot

SVG(model_to_dot(model).create(prog='dot', format='svg'))
```

Out[370]:



In [371]:
```
print(model.summary())
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense_106 (Dense)            (None, 269)               144991    
_________________________________________________________________
dropout_23 (Dropout)         (None, 269)               0         
_________________________________________________________________
dense_107 (Dense)            (None, 134)               36180     
_________________________________________________________________
dropout_24 (Dropout)         (None, 134)               0         
_________________________________________________________________
dense_108 (Dense)            (None, 75)                10125     
_________________________________________________________________
dense_109 (Dense)            (None, 1)                 76        
=================================================================
Total params: 191,372
Trainable params: 191,372
Non-trainable params: 0
_________________________________________________________________
None
```

In [477]:
```
y_test_pred = (model.predict(X_test)>0.4)
from sklearn.metrics import accuracy_score, precision_score, recall_score
Recall=recall_score(y_test,y_test_pred)
Precision=precision_score(y_test,y_test_pred)
ACC=accuracy_score(y_test,y_test_pred)
print("acu: {0:.2f}".format(ACC))
print("precision: {0:.2f}".format(Precision))
print("recall: {0:.2f}".format(Recall))
```

```
acu: 0.92
precision: 0.87
recall: 0.99
```

In [421]:
```
Fmeasure=((Recall*Precision)/((Recall+Precision)))*2
Fmeasure
```

Out[421]: 0.9269975463401599

In [ ]:
```
model.save('weights.model')
```

base line model

In [399]:
```
import numpy
from pandas import read_csv
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Dropout
from keras.wrappers.scikit_learn import KerasClassifier
from keras.constraints import maxnorm
from keras.optimizers import SGD
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import StratifiedKFold
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
# fix random seed for reproducibility
```

In [402]:
```
encoder = LabelEncoder()
encoder.fit(Y)
encoded_Y = encoder.transform(Y)
encoded_Y.shape
```

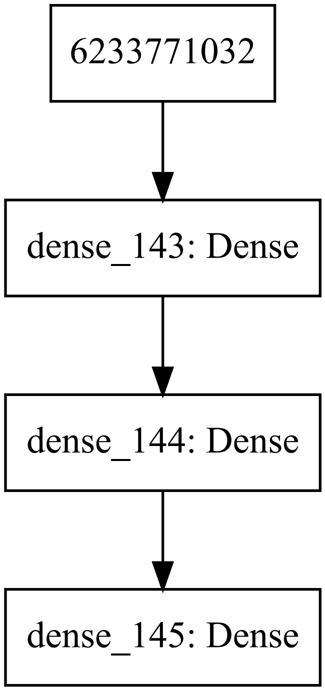
Out[402]: (70416,)

In [403]:
```
# baseline
def create_baseline():
   # create model
        model = Sequential()
        model.add(Dense(269, input_dim=538, kernel_initializer='normal', activ
ation='relu'))
        model.add(Dense(269, kernel_initializer='normal', activation='relu'))
        model.add(Dense(1, kernel_initializer='normal', activation='sigmoid'))
        # Compile model
        sgd = SGD(lr=0.01, momentum=0.8, decay=0.0, nesterov=False)
        model.compile(loss='binary_crossentropy', optimizer=sgd, metrics=['acc
uracy'])
        return model

seed = 7
numpy.random.seed(seed)
estimators = []
estimators.append(('standardize', StandardScaler()))
estimators.append(('mlp', KerasClassifier(build_fn=create_baseline, epochs=10,
batch_size=16, verbose=0)))
pipeline = Pipeline(estimators)
kfold = StratifiedKFold(n_splits=10, shuffle=True, random_state=seed)
results = cross_val_score(pipeline, X, encoded_Y, cv=kfold)
print("Baseline: %.2f%% (%.2f%%)" % (results.mean()*100, results.std()*100))
```

```
Baseline: 88.02% (0.58%)
```

In [404]:
```
SVG(model_to_dot(create_baseline()).create(prog='dot', format='svg'))
```

Out[404]:



In [ ]:

In [ ]:

In [ ]:

In [ ]:

refrences: https://www.kaggle.com/sid321axn/fraud-detection-deep-learning-with-smote/notebook (https://www.kaggle.com/sid321axn/fraud-detection-deep-learning-with-smote/notebook) https://medium.com/@klintcho/explaining-precision-and-recall-c770eb9c69e9 (https://medium.com/@klintcho/explaining-precision-and-recall-c770eb9c69e9) https://secml.github.io/class8/ (https://secml.github.io/class8/)

In [ ]: