NAME: AVIREDDY NVSRK ROHAN

REG.NO: 19BCE1180

DATASET USED : KDD-99

In [24]:
```
import pandas as pd
import numpy as np
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix,ConfusionMatrixDisplay
```

In [2]:
```
data=pd.read_csv("/content/corrected",header=None)
```

In [3]:
```
data.columns=['duration', 'protocol_type', 'service', 'flag', 'src_bytes',
        'dst_bytes', 'land', 'wrong_fragment', 'urgent', 'hot',
       'num_failed_logins', 'logged_in', 'num_compromised',
       'root_shell', 'su_attempted', 'num_root', 'num_file_creations',
       'num_shells', 'num_access_files', 'num_outbound_cmds',
       'is_host_login', 'is_guest_login', 'count', 'srv_count',
       'serror_rate', 'srv_serror_rate', 'rerror_rate',
       'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate',
       'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count',
       'dst_host_same_srv_rate', 'dst_host_diff_srv_rate',
       'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate',
       'dst_host_serror_rate', 'dst_host_srv_serror_rate',
       'dst_host_rerror_rate', 'dst_host_srv_rerror_rate', 'class']
```

In [28]:
```
data.head()
```

Out[28]:

	duration	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	nu
0	0	udp	private	SF	105	146	0	0	0	0	
1	0	udp	private	SF	105	146	0	0	0	0	
2	0	udp	private	SF	105	146	0	0	0	0	
3	0	udp	private	SF	105	146	0	0	0	0	
4	0	udp	private	SF	105	146	0	0	0	0	

In [17]:
```
data["class"].value_counts()
```

Out[17]:
```
smurf.              164091
```

```
normal.              60593
neptune.             58001
snmpgetattack.        7741
mailbomb.             5000
guess_passwd.         4367
snmpguess.            2406
satan.                1633
warezmaster.          1602
back.                 1098
mscan.                1053
apache2.               794
processtable.          759
saint.                 736
portsweep.             354
ipsweep.               306
httptunnel.            158
pod.                    87
nmap.                   84
buffer_overflow.        22
multihop.               18
named.                  17
sendmail.               17
ps.                     16
xterm.                  13
rootkit.                13
teardrop.               12
xlock.                   9
land.                    9
xsnoop.                  4
ftp_write.               3
udpstorm.                2
loadmodule.              2
phf.                     2
sqlattack.               2
worm.                    2
perl.                    2
imap.                    1
Name: class, dtype: int64
```

CLASS DISTRIBUTION FOR MALICIOUS AND NORMAL TYPES

In [19]:

```
print((data.shape[0]-60593)*100/data.shape[0])
print((60593)*100/data.shape[0])
```

```
80.51853685669182
19.48146314330818
```

CLASS DISTRIBUTION ACCROSS THE TARGET LABELS

In [35]:

```
data["class"].value_counts()*100/data.shape[0]
```

Out[35]:

```
smurf.              52.757460
normal.             19.481463
neptune.            18.648100
snmpgetattack.       2.488835
mailbomb.            1.607567
guess_passwd.        1.404049
snmpguess.           0.773561
satan.               0.525031
```

```
warezmaster.          0.515065
back.                 0.353022
mscan.                0.338554
apache2.              0.255282
processtable.         0.244029
saint.                0.236634
portsweep.            0.113816
ipsweep.              0.098383
httptunnel.           0.050799
pod.                  0.027972
nmap.                 0.027007
buffer_overflow.      0.007073
multihop.             0.005787
named.                0.005466
sendmail.             0.005466
ps.                   0.005144
xterm.                0.004180
rootkit.              0.004180
teardrop.             0.003858
xlock.                0.002894
land.                 0.002894
xsnoop.               0.001286
ftp_write.            0.000965
udpstorm.             0.000643
loadmodule.           0.000643
phf.                  0.000643
sqlattack.            0.000643
worm.                 0.000643
perl.                 0.000643
imap.                 0.000322
Name: class, dtype: float64
```

column for trying binary classification

In [7]:

```
bin_label=[]
for i in data["class"]:
  if i=="normal.":
    bin_label.append("normal")
  else:
    bin_label.append("attack")
data["bin_label"]=bin_label
```

In [29]:

```
data.describe()
```

Out[29]:

	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	311(
count	311029.000000	3.110290e+05	3.110290e+05	311029.000000	311029.000000	311029.000000	
mean	17.902736	1.731702e+03	7.479937e+02	0.000029	0.000762	0.000051	
std	407.644400	1.276567e+05	1.612018e+04	0.005379	0.040367	0.009821	
min	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	
25%	0.000000	1.050000e+02	0.000000e+00	0.000000	0.000000	0.000000	
50%	0.000000	5.200000e+02	0.000000e+00	0.000000	0.000000	0.000000	
75%	0.000000	1.032000e+03	0.000000e+00	0.000000	0.000000	0.000000	
max	57715.000000	6.282565e+07	5.203179e+06	1.000000	3.000000	3.000000	

In [34]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311029 entries, 0 to 311028
Data columns (total 43 columns):
 #   Column                       Non-Null Count   Dtype  
---  ------                       --------------   -----  
 0   duration                     311029 non-null  int64  
 1   protocol_type                311029 non-null  object 
 2   service                      311029 non-null  object 
 3   flag                         311029 non-null  object 
 4   src_bytes                    311029 non-null  int64  
 5   dst_bytes                    311029 non-null  int64  
 6   land                         311029 non-null  int64  
 7   wrong_fragment               311029 non-null  int64  
 8   urgent                       311029 non-null  int64  
 9   hot                          311029 non-null  int64  
 10  num_failed_logins            311029 non-null  int64  
 11  logged_in                    311029 non-null  int64  
 12  num_compromised              311029 non-null  int64  
 13  root_shell                   311029 non-null  int64  
 14  su_attempted                 311029 non-null  int64  
 15  num_root                     311029 non-null  int64  
 16  num_file_creations           311029 non-null  int64  
 17  num_shells                   311029 non-null  int64  
 18  num_access_files             311029 non-null  int64  
 19  num_outbound_cmds            311029 non-null  int64  
 20  is_host_login                311029 non-null  object 
 21  is_guest_login               311029 non-null  object 
 22  count                        311029 non-null  int64  
 23  srv_count                    311029 non-null  int64  
 24  serror_rate                  311029 non-null  float64
 25  srv_serror_rate              311029 non-null  float64
 26  rerror_rate                  311029 non-null  float64
 27  srv_rerror_rate              311029 non-null  float64
 28  same_srv_rate                311029 non-null  float64
 29  diff_srv_rate                311029 non-null  float64
 30  srv_diff_host_rate           311029 non-null  float64
 31  dst_host_count               311029 non-null  int64  
 32  dst_host_srv_count           311029 non-null  int64  
 33  dst_host_same_srv_rate       311029 non-null  float64
 34  dst_host_diff_srv_rate       311029 non-null  float64
 35  dst_host_same_src_port_rate  311029 non-null  float64
 36  dst_host_srv_diff_host_rate  311029 non-null  float64
 37  dst_host_serror_rate         311029 non-null  float64
 38  dst_host_srv_serror_rate     311029 non-null  float64
 39  dst_host_rerror_rate         311029 non-null  float64
 40  dst_host_srv_rerror_rate     311029 non-null  float64
 41  class                        311029 non-null  object 
 42  bin_label                    311029 non-null  object 
dtypes: float64(15), int64(21), object(7)
memory usage: 102.0+ MB
```

In [4]:
```
data["is_host_login"]=data["is_host_login"].astype("object")
data["is_guest_login"]=data["is_guest_login"].astype("object")
```

NORMALIZING THE DATA

In [5]:
```
le=LabelEncoder()
data["protocol_type"]=le.fit_transform(data["protocol_type"])
data["service"]=le.fit_transform(data["service"])
data["flag"]=le.fit_transform(data["flag"])
```

In [8]:
```
multi_label=data["class"]
bin_label=data["bin_label"]
data.drop(["class","bin_label"],inplace=True,axis=1)
```

In [9]:
```
multi_label=pd.Series(multi_label)
bin_label=pd.Series(bin_label)
```

In [10]:
```
st_obj=StandardScaler()
data_std=st_obj.fit_transform(data)
```

In [11]:
```
data_std=pd.DataFrame(data_std)
```

In [12]:
```
data_std
```

Out[12]:

	0	1	2	3	4	5	6	7	8	
0	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037344	-0.005379	-0.018876	-0.005238	-0
1	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037344	-0.005379	-0.018876	-0.005238	-0
2	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037344	-0.005379	-0.018876	-0.005238	-0
3	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037344	-0.005379	-0.018876	-0.005238	-0
4	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037344	-0.005379	-0.018876	-0.005238	-0
...	...	...	...	...	...	...	...	...	...	
311024	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037282	-0.005379	-0.018876	-0.005238	-0
311025	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037282	-0.005379	-0.018876	-0.005238	-0
311026	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037282	-0.005379	-0.018876	-0.005238	-0
311027	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037282	-0.005379	-0.018876	-0.005238	-0
311028	-0.043918	2.232618	1.428547	0.47954	-0.012743	-0.037282	-0.005379	-0.018876	-0.005238	-0

311029 rows × 41 columns

In [13]:
```
import imblearn
from collections import Counter
from imblearn.over_sampling import RandomOverSampler
from imblearn.under_sampling import RandomUnderSampler
from imblearn.over_sampling import SMOTE
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/externals/six.py:31: FutureWarning: The m
```

```
odule is deprecated in version 0.21 and will be removed in version 0.23 since we've drop
ped support for Python 2.7. Please rely on the official version of six (https://pypi.or
g/project/six/).
  "(https://pypi.org/project/six/).", FutureWarning)
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:144: FutureWarning:
The sklearn.neighbors.base module is  deprecated in version 0.22 and will be removed in
version 0.24. The corresponding classes / functions should instead be imported from skle
arn.neighbors. Anything that cannot be imported from sklearn.neighbors is now part of th
e private API.
  warnings.warn(message, FutureWarning)
```

In [28]:
```
from matplotlib import pyplot as plt
```

imbalanced data

In [42]:
```
X_train,X_test,y_train,y_test=train_test_split(data_std,bin_label,random_state=1,test_s
```

Randomly Oversampled data

In [15]:
```
oversample = RandomOverSampler(sampling_strategy='minority')
X_over, y_over = oversample.fit_resample(data_std,bin_label)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
unction safe_indexing is deprecated; safe_indexing is deprecated in version 0.22 and wil
l be removed in version 0.24.
  warnings.warn(msg, category=FutureWarning)
```

In [54]:
```
print(Counter(y_over))
```

```
Counter({'normal': 250436, 'attack': 250436})
```

70 30 split

In [16]:
```
X_train_ovr,X_test_ovr,y_train_ovr,y_test_ovr=train_test_split(X_over,y_over,random_sta
```

Randomly Undersampled data

In [17]:
```
undersample = RandomUnderSampler(sampling_strategy='majority')
X_under,y_under=undersample.fit_resample(data_std,bin_label)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
unction safe_indexing is deprecated; safe_indexing is deprecated in version 0.22 and wil
l be removed in version 0.24.
  warnings.warn(msg, category=FutureWarning)
```

In [57]:
```
print(Counter(y_under))
```

```
Counter({'attack': 60593, 'normal': 60593})
```

In [18]:
```
X_train_under,X_test_under,y_train_under,y_test_under=train_test_split(X_over,y_over,ra
```

SMOTE data

In [19]:
```
smote=SMOTE()
X_smote,y_smote=smote.fit_resample(data_std,bin_label)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: F
unction safe_indexing is deprecated; safe_indexing is deprecated in version 0.22 and wil
l be removed in version 0.24.
  warnings.warn(msg, category=FutureWarning)
```

In [62]:
```
X_smote
```

Out[62]:
```
array([[-0.0439176 ,  2.23261817,  1.42854674, ..., -0.25232616,
        -0.4142494 , -0.40884161],
       [-0.0439176 ,  2.23261817,  1.42854674, ..., -0.25232616,
        -0.4142494 , -0.40884161],
       [-0.0439176 ,  2.23261817,  1.42854674, ..., -0.25232616,
        -0.4142494 , -0.40884161],
       ...,
       [-0.0439176 ,  2.23261817,  1.42854674, ..., -0.25232616,
        -0.4142494 , -0.40884161],
       [-0.0439176 ,  0.6870657 , -0.42228509, ..., -0.25232616,
        -0.4142494 , -0.40884161],
       [-0.0439176 ,  0.6870657 , -0.21663711, ..., -0.25232616,
        -0.4142494 , -0.40884161]])
```

In [61]:
```
print(Counter(y_smote))
```

```
Counter({'normal': 250436, 'attack': 250436})
```

In [20]:
```
X_train_smote,X_test_smote,y_train_smote,y_test_smote=train_test_split(X_over,y_over,ra
```

MODEL

In [46]:
```
lr=LogisticRegression(C=100.0, random_state=1, solver='lbfgs',max_iter=400)
```

performance on imbalanced data

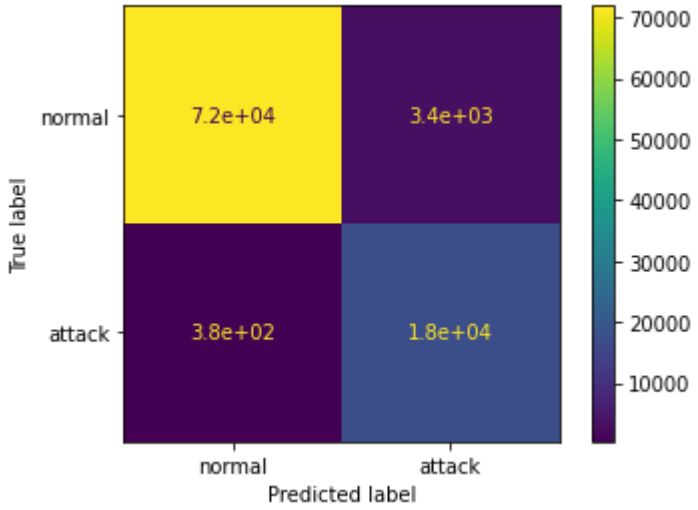
In [47]:
```
lr.fit(X_train,y_train)
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/linear_model/_logistic.py:940: Convergenc
eWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
```

Out[47]:
```
LogisticRegression(C=100.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=400,
                   multi_class='auto', n_jobs=None, penalty='l2',
```

```
                   random_state=1, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [48]:
```
pred=lr.predict(X_test)
cm = confusion_matrix(y_test,pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=["normal","attack"])
disp.plot()
plt.show()
```



In [50]:
```
print(classification_report(y_test,pred))
```

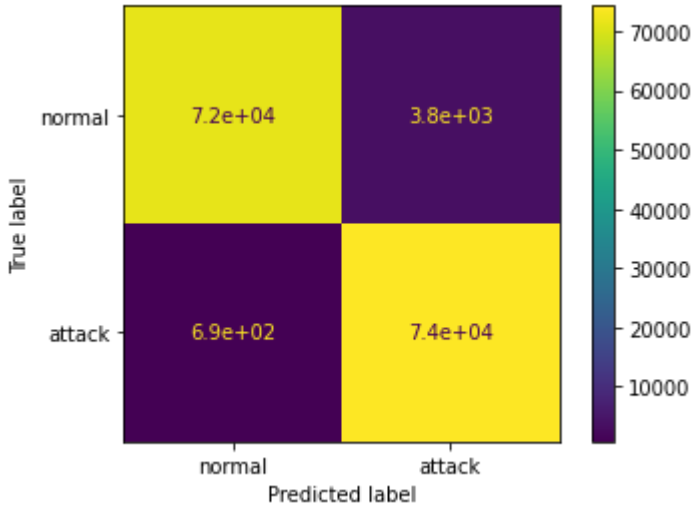
```
              precision    recall  f1-score   support

      attack       0.99      0.96      0.97     75289
      normal       0.84      0.98      0.90     18020

    accuracy                           0.96     93309
   macro avg       0.92      0.97      0.94     93309
weighted avg       0.96      0.96      0.96     93309
```

performance on undersampled data

In [26]:
```
lr.fit(X_train_under,y_train_under)
```

Out[26]:
```
LogisticRegression(C=100.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=300,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=1, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [29]:
```
pred_under=lr.predict(X_test_under)
cm = confusion_matrix(y_test_under,pred_under)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=["normal","attack"])
disp.plot()
plt.show()
```



In [36]:
```
print(classification_report(y_test_under,pred_under))
```

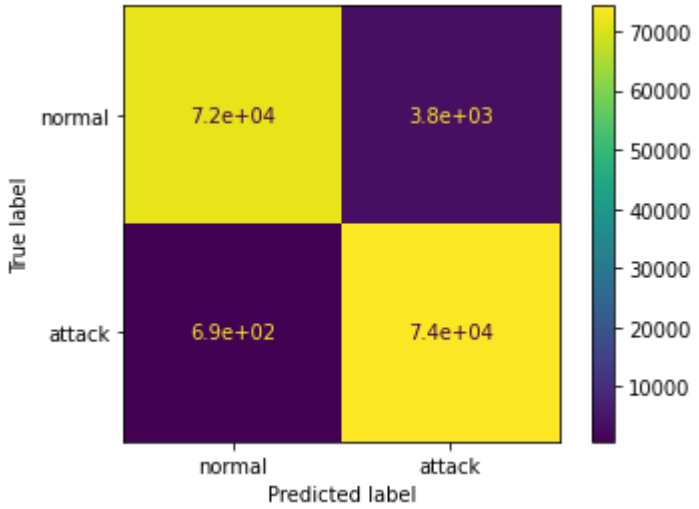
```
              precision    recall  f1-score   support

      attack       0.99      0.95      0.97     75284
      normal       0.95      0.99      0.97     74978

    accuracy                           0.97    150262
   macro avg       0.97      0.97      0.97    150262
weighted avg       0.97      0.97      0.97    150262
```

performance on oversampled data

In [34]:
```
lr.fit(X_train_ovr,y_train_ovr)
```

Out[34]:
```
LogisticRegression(C=100.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=300,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=1, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [38]:
```
pred_over=lr.predict(X_test_ovr)
cm = confusion_matrix(y_test_ovr,pred_over)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=["normal","attack"])
disp.plot()
plt.show()
```



In [39]:
```
print(classification_report(y_test_ovr,pred_over))
```

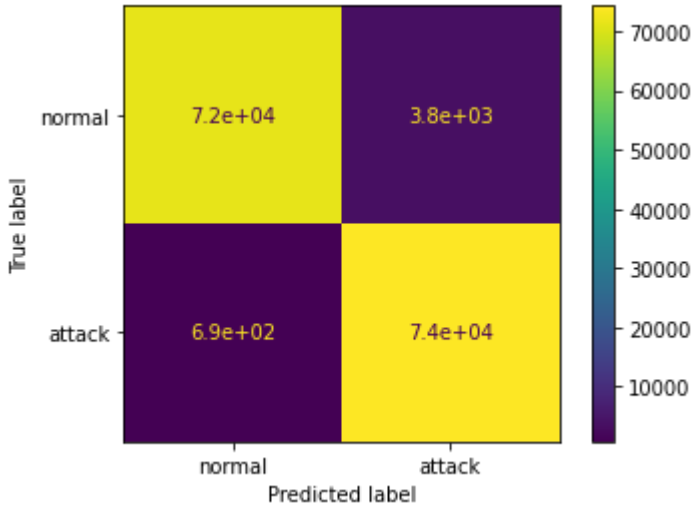
```
              precision    recall  f1-score   support

      attack       0.99      0.95      0.97     75284
      normal       0.95      0.99      0.97     74978

    accuracy                           0.97    150262
   macro avg       0.97      0.97      0.97    150262
weighted avg       0.97      0.97      0.97    150262
```

performance on SMOTE data

In [32]:
```
lr.fit(X_train_smote,y_train_smote)
```

Out[32]:
```
LogisticRegression(C=100.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=300,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=1, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [33]:
```
pred_smote=lr.predict(X_test_smote)
cm = confusion_matrix(y_test_smote,pred_smote)
disp = ConfusionMatrixDisplay(confusion_matrix=cm,display_labels=["normal","attack"])
disp.plot()
plt.show()
```



In [41]:
```
print(classification_report(y_test_smote,pred_smote))
```

```
              precision    recall  f1-score   support

      attack       0.99      0.95      0.97     75284
      normal       0.95      0.99      0.97     74978

    accuracy                           0.97    150262
   macro avg       0.97      0.97      0.97    150262
weighted avg       0.97      0.97      0.97    150262
```