In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [7]:

```
data = pd.read_csv('/cxldata/projects/creditcard.csv')
data.head(10)
```

Out[7]:

	Time	V1	V2	V3	V4	V5	V6	V7	V
0	0.0	-1.359807	-0.072781	2.536347	1.378155	-0.338321	0.462388	0.239599	0.09869
1	0.0	1.191857	0.266151	0.166480	0.448154	0.060018	-0.082361	-0.078803	0.08510
2	1.0	-1.358354	-1.340163	1.773209	0.379780	-0.503198	1.800499	0.791461	0.24767
3	1.0	-0.966272	-0.185226	1.792993	-0.863291	-0.010309	1.247203	0.237609	0.37743
4	2.0	-1.158233	0.877737	1.548718	0.403034	-0.407193	0.095921	0.592941	-0.27053
5	2.0	-0.425966	0.960523	1.141109	-0.168252	0.420987	-0.029728	0.476201	0.26031
6	4.0	1.229658	0.141004	0.045371	1.202613	0.191881	0.272708	-0.005159	0.08121
7	7.0	-0.644269	1.417964	1.074380	-0.492199	0.948934	0.428118	1.120631	-3.80786
8	7.0	-0.894286	0.286157	-0.113192	-0.271526	2.669599	3.721818	0.370145	0.85108
9	9.0	-0.338262	1.119593	1.044367	-0.222187	0.499361	-0.246761	0.651583	0.06953

10 rows × 31 columns

In [8]:

```
data.shape
```

Out[8]:

```
(284807, 31)
```

In [12]:

```
data.describe()
```

Out[12]:

	Time	V1	V2	V3	V4	
count	284807.000000	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e+05	2.848070e
mean	94813.859575	3.919560e-15	5.688174e-16	-8.769071e-15	2.782312e-15	-1.552563e
std	47488.145955	1.958696e+00	1.651309e+00	1.516255e+00	1.415869e+00	1.380247e
min	0.000000	-5.640751e+01	-7.271573e+01	-4.832559e+01	-5.683171e+00	-1.137433e
25%	54201.500000	-9.203734e-01	-5.985499e-01	-8.903648e-01	-8.486401e-01	-6.915971e
50%	84692.000000	1.810880e-02	6.548556e-02	1.798463e-01	-1.984653e-02	-5.433583e
75%	139320.500000	1.315642e+00	8.037239e-01	1.027196e+00	7.433413e-01	6.119264e
max	172792.000000	2.454930e+00	2.205773e+01	9.382558e+00	1.687534e+01	3.480167e

8 rows × 31 columns

In [13]:

```
data.isnull().sum()
```

Out[13]:

```
Time      0
V1        0
V2        0
V3        0
V4        0
V5        0
V6        0
V7        0
V8        0
V9        0
V10       0
V11       0
V12       0
V13       0
V14       0
V15       0
V16       0
V17       0
V18       0
V19       0
V20       0
V21       0
V22       0
V23       0
V24       0
V25       0
V26       0
V27       0
V28       0
Amount    0
Class     0
dtype: int64
```

In [14]:

```
X=data.loc[:,data.columns!='Class']
```

In [15]:

```
y=data.loc[:,data.columns=='Class']
```

In [16]:

```
print(data['Class'].value_counts())
```

```
0    284315
1       492
Name: Class, dtype: int64
```

In [17]:

```
print('Valid Transactions: ', round(data['Class'].value_counts()[0]/len(data) * 100,2),
'% of the dataset')

print('Fraudulent Transactions: ', round(data['Class'].value_counts()[1]/len(data) * 10
0,2), '% of the dataset')
```

```
Valid Transactions:  99.83 % of the dataset
Fraudulent Transactions:  0.17 % of the dataset
```

In [20]:

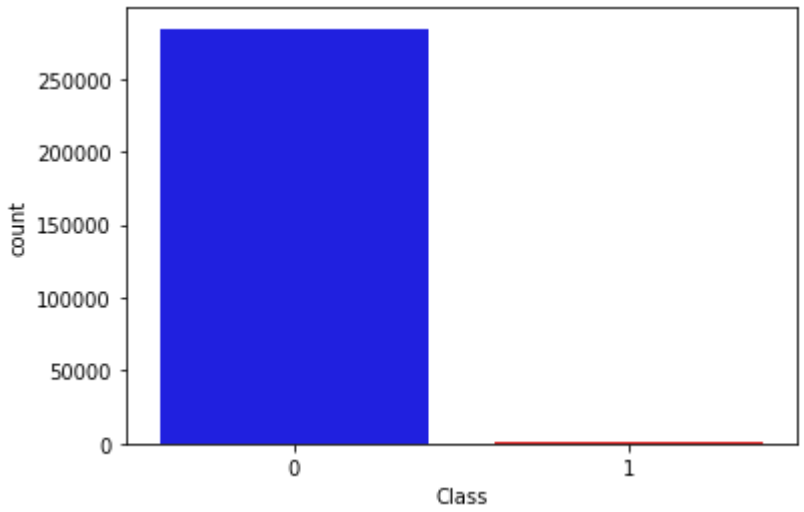
```
colors = ['blue','red']
```

In [21]:

```
sns.countplot('Class', data=data, palette=colors)
```

Out[21]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe245ba6cc0>
```



In [23]:

```
from sklearn.model_selection import train_test_split
```

In [24]:

```
X_train, X_test, y_train, y_test =  train_test_split(X, y, test_size=0.3, random_state=
0)
```

In [25]:

```
print("Transactions in X_train dataset: ", X_train.shape)
print("Transaction classes in y_train dataset: ", y_train.shape)

print("Transactions in X_test dataset: ", X_test.shape)
print("Transaction classes in y_test dataset: ", y_test.shape)
```

```
Transactions in X_train dataset:  (199364, 30)
Transaction classes in y_train dataset:  (199364, 1)
Transactions in X_test dataset:  (85443, 30)
Transaction classes in y_test dataset:  (85443, 1)
```

In [31]:

```
from sklearn.preprocessing import StandardScaler
```

In [32]:

```
scaler_amount = StandardScaler()
scaler_time = StandardScaler()
```

In [33]:

```
X_train['normAmount'] = scaler_amount .fit_transform(X_train['Amount'].values.reshape(-
1, 1))
```

```
/usr/local/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:1: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [34]:

```
X_test['normAmount'] = scaler_amount .transform(X_test['Amount'].values.reshape(-1, 1))
```

```
/usr/local/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:1: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [35]:

```
X_train['normTime'] = scaler_time .fit_transform(X_train['Time'].values.reshape(-1, 1))
```

```
/usr/local/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:1: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [36]:

```
X_test['normTime'] = scaler_time .transform(X_test['Time'].values.reshape(-1, 1))
```

```
/usr/local/anaconda/lib/python3.6/site-packages/ipykernel_launcher.py:1: S
ettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-doc
s/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [37]:

```
X_train = X_train.drop(['Time', 'Amount'], axis=1)
X_test = X_test.drop(['Time', 'Amount'], axis=1)
```

In [40]:

```
X_train.head()
```

Out[40]:

	V1	V2	V3	V4	V5	V6	V7	V8
161145	-0.132066	0.107044	-0.650588	-0.996032	1.814333	1.740740	0.496852	0.633016
204520	2.125994	0.014207	-1.514760	0.115021	0.598510	-0.333235	0.199289	-0.264353
182659	-0.086694	0.166240	1.573127	0.687266	0.222359	1.102606	1.575093	-1.098608
25117	1.352339	-0.534984	0.555143	-0.629355	-1.144170	-0.852967	-0.642128	-0.032659
227642	-1.526760	0.647782	0.615391	-0.561114	0.836950	-0.514251	0.984325	-0.097430

5 rows × 30 columns

In [48]:

```
from imblearn.over_sampling import SMOTE
```

```
Using TensorFlow backend.
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:516: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:517: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:518: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:519: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:520: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorflow/python/framewor
k/dtypes.py:525: FutureWarning: Passing (type, 1) or '1type' as a synonym
of type is deprecated; in a future version of numpy, it will be understood
as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:541: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
erstood as (type, (1,)) / '(1,)type'.
  _np_qint8 = np.dtype([("qint8", np.int8, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:542: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
erstood as (type, (1,)) / '(1,)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:543: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
erstood as (type, (1,)) / '(1,)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:544: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
erstood as (type, (1,)) / '(1,)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:545: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
erstood as (type, (1,)) / '(1,)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
/usr/local/anaconda/lib/python3.6/site-packages/tensorboard/compat/tensorf
low_stub/dtypes.py:550: FutureWarning: Passing (type, 1) or '1type' as a s
ynonym of type is deprecated; in a future version of numpy, it will be und
```

```
erstood as (type, (1,)) / '(1,)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
```

In [49]:

```
print("Before over-sampling:\n", y_train['Class'].value_counts())
```

```
Before over-sampling:
 0    199019
1       345
Name: Class, dtype: int64
```

In [50]:

```
sm = SMOTE()
```

In [51]:

```
X_train_res, y_train_res = sm.fit_sample(X_train, y_train['Class'])
```

In [52]:

```
print("After over-sampling:\n", y_train_res.value_counts())
```

```
After over-sampling:
 1    199019
0    199019
Name: Class, dtype: int64
```

In [57]:

```
from sklearn.model_selection import GridSearchCV
```

In [58]:

```
from sklearn.linear_model import LogisticRegression
```

In [59]:

```
from sklearn.metrics import confusion_matrix, auc, roc_curve
```

In [60]:

```
parameters = {"penalty": ['l1', 'l2'], 'C': [0.001, 0.01, 0.1, 1, 10, 100, 1000]}
```

In [61]:

```
lr = LogisticRegression()
```

In [62]:

```
clf = GridSearchCV(lr, parameters, cv=5, verbose=5, n_jobs=3)
```

In [63]:

```
k = clf.fit(X_train_res, y_train_res)
```

```
Fitting 5 folds for each of 14 candidates, totalling 70 fits

[Parallel(n_jobs=3)]: Using backend LokyBackend with 3 concurrent workers.
[Parallel(n_jobs=3)]: Done  12 tasks      | elapsed:   14.3s
[Parallel(n_jobs=3)]: Done  70 out of  70 | elapsed:  3.2min finished
```

In [64]:

```
print(k.best_params_)
```

```
{'C': 10, 'penalty': 'l2'}
```

In [73]:

```
lr_gridcv_best = clf.best_estimator_
```

In [74]:

```
y_test_pre = lr_gridcv_best.predict(X_test)
```

In [75]:

```
cnf_matrix_test = confusion_matrix(y_test, y_test_pre)
```

In [76]:

```
print("Recall metric in the test dataset:", (cnf_matrix_test[1,1]/(cnf_matrix_test[1,0]
+cnf_matrix_test[1,1] )))
```

```
Recall metric in the test dataset: 0.9183673469387755
```

In [77]:

```
y_train_pre = lr_gridcv_best.predict(X_train_res)
```

In [78]:

```
cnf_matrix_train = confusion_matrix(y_train_res, y_train_pre)
```

In [79]:

```
print("Recall metric in the train dataset:", (cnf_matrix_train[1,1]/(cnf_matrix_train[1
,0]+cnf_matrix_train[1,1] )))
```

```
Recall metric in the train dataset: 0.9196810354790246
```

In [85]:

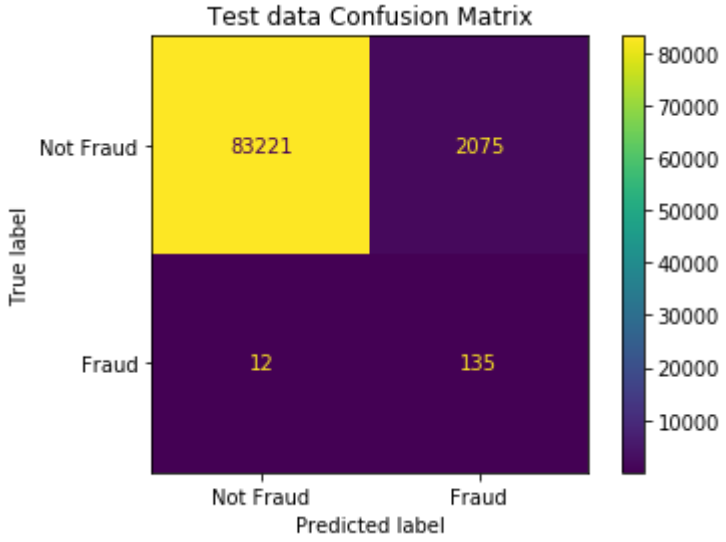
```
from sklearn.metrics import plot_confusion_matrix
```

In [86]:

```
class_names = ['Not Fraud', 'Fraud']
```

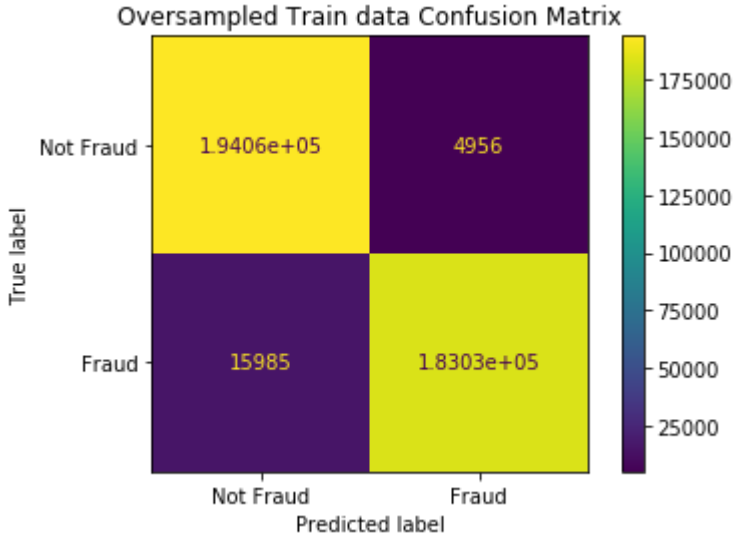
In [87]:

```
plot_confusion_matrix(k, X_test, y_test,  values_format = '.5g', display_labels=class_n
ames)
plt.title("Test data Confusion Matrix")
plt.show()
```



In [88]:

```
plot_confusion_matrix(k, X_train_res, y_train_res,  values_format = '.5g', display_labe
ls=class_names)
plt.title("Oversampled Train data Confusion Matrix")
plt.show()
```



In [91]:

```
y_k =  k.decision_function(X_test)
```

In [92]:

```
fpr, tpr, thresholds = roc_curve(y_test, y_k)
```

In [93]:

```
roc_auc = auc(fpr, tpr)
```

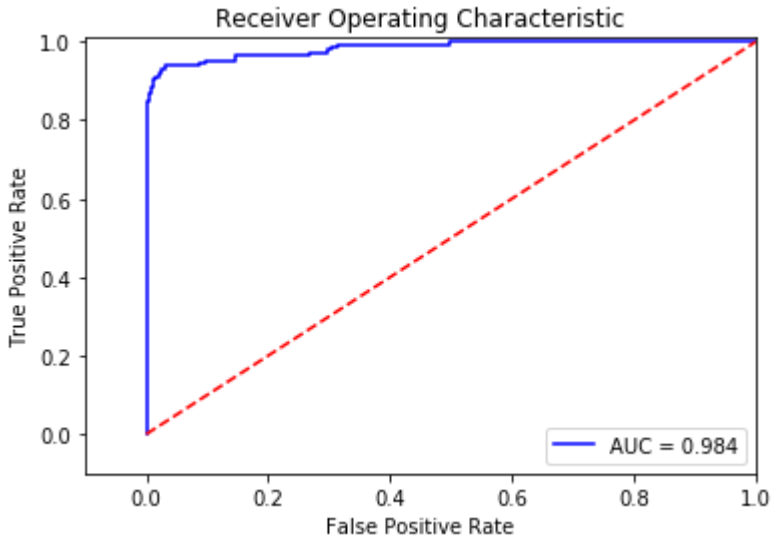
In [94]:

```
print("ROC-AUC:", roc_auc)
```

```
ROC-AUC: 0.9839701074577271
```

In [95]:

```
plt.title('Receiver Operating Characteristic')
plt.plot(fpr, tpr, 'b',label='AUC = %0.3f'% roc_auc)
plt.legend(loc='lower right')
plt.plot([0,1],[0,1],'r--')
plt.xlim([-0.1,1.0])
plt.ylim([-0.1,1.01])
plt.ylabel('True Positive Rate')
plt.xlabel('False Positive Rate')
plt.show()
```



In [ ]: