In [1]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

In [2]:

```
# Loading dataset
input_data = pd.read_csv("creditcard.csv")

# Droping 'Amount'
input_data = input_data.drop(['Time', 'Amount'],axis=1)
```

In [4]:

```
# Create X and y
y = input_data['Class']
X = input_data.drop(['Class'], axis=1)
```

In [5]:

```
# Shuffle and split the data into training and testing subsets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size =
0.4, random_state = 0)
```

In [6]:

```
# RandomForestClassifier/ Decision Trees
from sklearn.ensemble import RandomForestClassifier

rf_classifier = RandomForestClassifier()
rf_classifier = rf_classifier.fit(X_train, y_train)

y_pred1 = rf_classifier.predict(X_train)

# calculating accuracy score
rf_classifier.score(X_test, y_test)

# calculating precision and recall scores
precision = precision_score(y_train, y_pred1, average='binary')
recall = recall_score(y_train, y_pred1, average='binary')

print(precision)
print(recall)

# generating confusion matrix
cm3 = confusion_matrix(y_train,y_pred1)

df_cm3 = pd.DataFrame(cm3, index = ['True (positive)', 'True (negativ
e)'])
df_cm3.columns = ['Predicted (positive)', 'Predicted (negative)']

sns.heatmap(df_cm3, annot=True, fmt="d")
```

```
1.0
0.962457337884
```

Out[6]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x104387b70>
```

```
In [7]: # Logistic Regression Classifier
        from sklearn.linear_model import LogisticRegression

        lm_classifier = LogisticRegression()
        lm_classifier.fit(X,y)
        y_pred3 = lm_classifier.predict(X_train)

        # calculating accuracy score
        lm_classifier.score(X_test, y_test)

        # calculating precision and recall score
        precision = precision_score(y_train, y_pred3, average='binary')
        recall = recall_score(y_train, y_pred3, average='binary')

        print(precision)
        print(recall)

        # generating confusion matrix
        cm2 = confusion_matrix(y_train,y_pred3)

        df_cm2 = pd.DataFrame(cm2, index = ['True(positive)',
        'True(negative)'])
        df_cm2.columns = ['Predicted (positive)', 'Predicted (negative)']

        sns.heatmap(df_cm2, annot=True, fmt="d")
```

```
        0.868544600939
        0.631399317406
```

```
Out[7]: <matplotlib.axes._subplots.AxesSubplot at 0x104387b70>
```

```
In [ ]:
```