The IMDB dataset actually comes packaged with keras and its allready tokenized, menaing the text is allready
converted in a sequence of unique word indices. The IMDB dataset contains 50,000 movie reviews (25,000 fo
training and 25,000 for testing). Each set contains of 50% positive and 50% negative reviews (12,500 x 2).

```
import numpy as np
from keras.datasets import imdb
import matplotlib.pyplot as plt
```

```
vocabulary=7500
```

Below code block has been commented out because software does not work any more

```
# save np.load
#np_load_old = np.load

# modify the default parameters of np.load
#np.load = lambda *a,**k: np_load_old(*a, allow_pickle=True, **k)

# call load_data with allow_pickle implicitly set to true
#(train_data, train_labels), (test_data, test_labels) = imdb.load_data(num_words=vocab

# restore np.load for future normal usage
#np.load = np_load_old
```

New code fix

```
np.load.__defaults__=(None, True, True, 'ASCII')
(train_data, train_labels), (test_data, test_labels) = imdb.load_data(num_words=vocabu
np.load.__defaults__=(None, False, True, 'ASCII')
```

```
[1, 14, 22, 16, 43, 530, 973, 1622, 1385, 65, 458, 4468, 66, 3941, 4, 173, 3
```

```
print(train_data[0])
```

```python
print(type(train_data[0]))
```

```python
def vectorize_sequences(sequences, dimension=vocabulary):
    results=np.zeros((len(sequences), dimension))
    for i, sequence in enumerate(sequences):
        results[i, sequence]=1
    return results
```

```python
x_train=vectorize_sequences(train_data)
x_test=vectorize_sequences(test_data)

y_train=np.asarray(train_labels).astype('float32')
y_test=np.asarray(test_labels).astype('float32')
```

```python
print(x_train[0])
```

```python
print(type(x_train[0]))
```

```python
print(type(y_train[0]))
```

```python
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(x_train, y_train)
score = model.score(x_test, y_test)
print("Accuracy:", score)
```

```python
from sklearn.metrics import confusion_matrix
y_pred=model.predict(x_test)
confusion_matrix(y_test,y_pred)
```

```python
from sklearn.metrics import roc_curve
y_pred_proba=model.predict_proba(x_test)[:,1]
fpr, tpr, thresholds=roc_curve(y_test, y_pred_proba)
```

```python
plt.plot([0,1], [0,1], '')
plt.plot(fpr, tpr, label='')
plt.xlabel('fpr')
plt.ylabel('tpr')
plt.title('NB MultiN ROC curve')
plt.show()
```

```python
from sklearn.metrics import roc_auc_score #area under the ROC curve
roc_auc_score(y_test, y_pred_proba)
```