# Logistic regression model using title data vectorized with TF-IDF

In [ ]:
```
#import packages

import numpy as np

import pandas as pd

from sklearn.model_selection import KFold, cross_validate

from sklearn.pipeline import Pipeline

from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, ConfusionMatrixDispl

import matplotlib as plt

from time import process_time
```

In [ ]:
```
#import data

%store -r x_title_train_tf
%store -r x_title_test_tf
%store -r y_title_train_tf
%store -r y_title_test_tf
```

In [ ]:
```
#rename variables for ease of use

x_train = x_title_train_tf
x_test = x_title_test_tf
y_train = y_title_train_tf
y_test = y_title_test_tf
```

In [ ]:
```
#define model

lr = LogisticRegression()
```

In [ ]:
```
#define scoring metrics for cross validation

scorer = {'accuracy': make_scorer(accuracy_score),
          'precision': make_scorer(precision_score),
          'recall': make_scorer(recall_score),
          'f1_score' : make_scorer(f1_score)
          }
```

In [ ]:
```
#define KFold

k_folds = KFold(n_splits = 5, random_state=42, shuffle=True)
```

In [ ]:
```
#cross validate on training set to check model stability

cv_scores_tfidf_title_lr = cross_validate(lr, x_train, y_train, cv = k_folds, scoring=scorer)
```

In [ ]:
```
#check cross validation scores

cv_scores_tfidf_title_lr
```

Out[ ]:
```
{'fit_time': array([0.28822947, 0.17752671, 0.24634099, 0.28224421, 0.18055081]),
 'score_time': array([0.01595712, 0.01097012, 0.01097059, 0.01097345, 0.0169487 ]),
 'test_accuracy': array([0.88058063, 0.884606  , 0.87983409, 0.87434427, 0.88300598]),
 'test_precision': array([0.87388393, 0.87619584, 0.86975964, 0.86783528, 0.87240022]),
 'test_recall': array([0.85597158, 0.86022099, 0.85706417, 0.84893849, 0.86007204]),
 'test_f1_score': array([0.86483501, 0.86813493, 0.86336524, 0.85828288, 0.86619227])}
```

In [ ]:
```
#fit model on the whole training set
start = process_time()

lr.fit(x_train, y_train)

end=process_time()
```

In [ ]:
```
#test model on test set

y_pred = lr.predict(x_test)
```

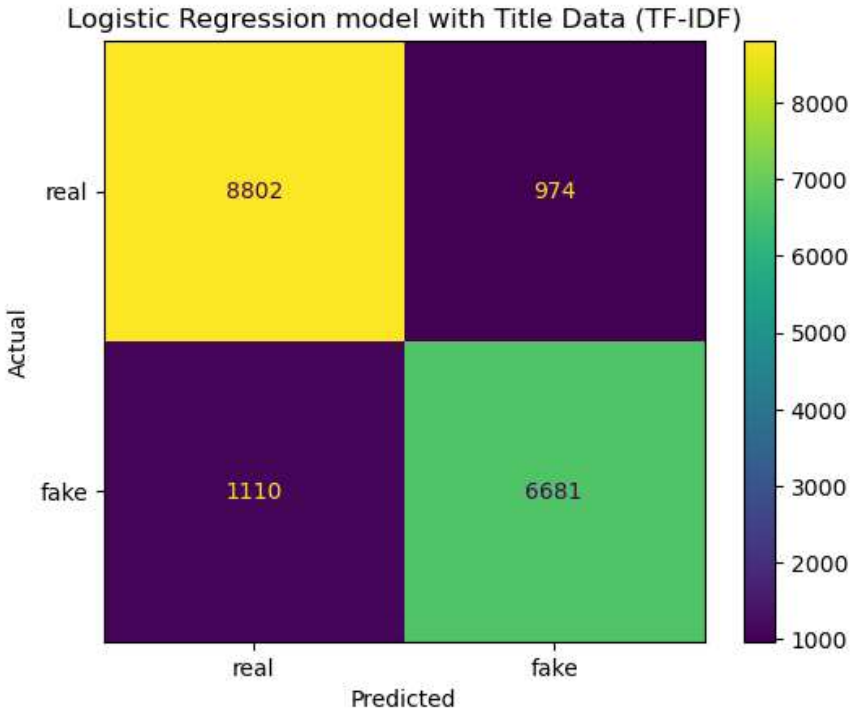
In [ ]:
```
#view classification report

print(classification_report(y_pred, y_test))
```

```
              precision    recall  f1-score   support

           0       0.90      0.89      0.89      9912
           1       0.86      0.87      0.87      7655

    accuracy                           0.88     17567
   macro avg       0.88      0.88      0.88     17567
weighted avg       0.88      0.88      0.88     17567
```

In [ ]:
```
#view confusion matrix

conf_matrix=confusion_matrix(y_test, y_pred)
```

In [ ]:
```
cm_plot = ConfusionMatrixDisplay(conf_matrix, display_labels = ['real', 'fake'])
cm_plot.plot(values_format='')
cm_plot.ax_.set(
                title='Logistic Regression model with Title Data (TF-IDF)',
                xlabel='Predicted',
                ylabel='Actual')
```

Out[ ]:
```
[Text(0.5, 1.0, 'Logistic Regression model with Title Data (TF-IDF)'),
 Text(0.5, 0, 'Predicted'),
 Text(0, 0.5, 'Actual')]
```




In [ ]:
```
accuracy=accuracy_score(y_test, y_pred)
precision=precision_score(y_test, y_pred)
recall=recall_score(y_test, y_pred)
f1=f1_score(y_test, y_pred)
fit_time=end-start
```

In [ ]:
```
dict = {'data type': 'title',
        'model type':'logistic regression',
        'vectorize type':'TF-IDF',
        'accuracy': accuracy,
        'precision': precision,
        'recall': recall,
        'f1': f1,
        'fit time': fit_time
       }
```

In [ ]:
```
tfidf_title_lr=pd.DataFrame.from_dict([dict])
```

In [ ]:
```
tfidf_title_lr
```

Out[ ]:

| | data type | model type | vectorize type | accuracy | precision | recall | f1 | fit time |
|---|---|---|---|---|---|---|---|---|
| **0** | title | logistic regression | TF-IDF | 0.881368 | 0.872763 | 0.857528 | 0.865078 | 0.546875 |

In [ ]:
```
#save results for later use

%store tfidf_title_lr
```

```
Stored 'tfidf_title_lr' (DataFrame)
```