ENRON POI CLASSIFICATION

MACHINE LEARNING CS 6140

ASHAY PANCHAL

ABOUT ENRON

Enron Corporation was an American energy, commodities, and services company based in Houston, Texas. It was founded by Kenneth Lay in 1985 as a merger between Lay's Houston Natural Gas and InterNorth, both relatively small regional companies. Before its bankruptcy on December 2, 2001, Enron employed approximately 20,600 staff and was a major electricity, natural gas, communications, and pulp and paper company, with claimed revenues of nearly $101 billion during 2000. Fortune named Enron "America's Most Innovative Company" for six consecutive years.

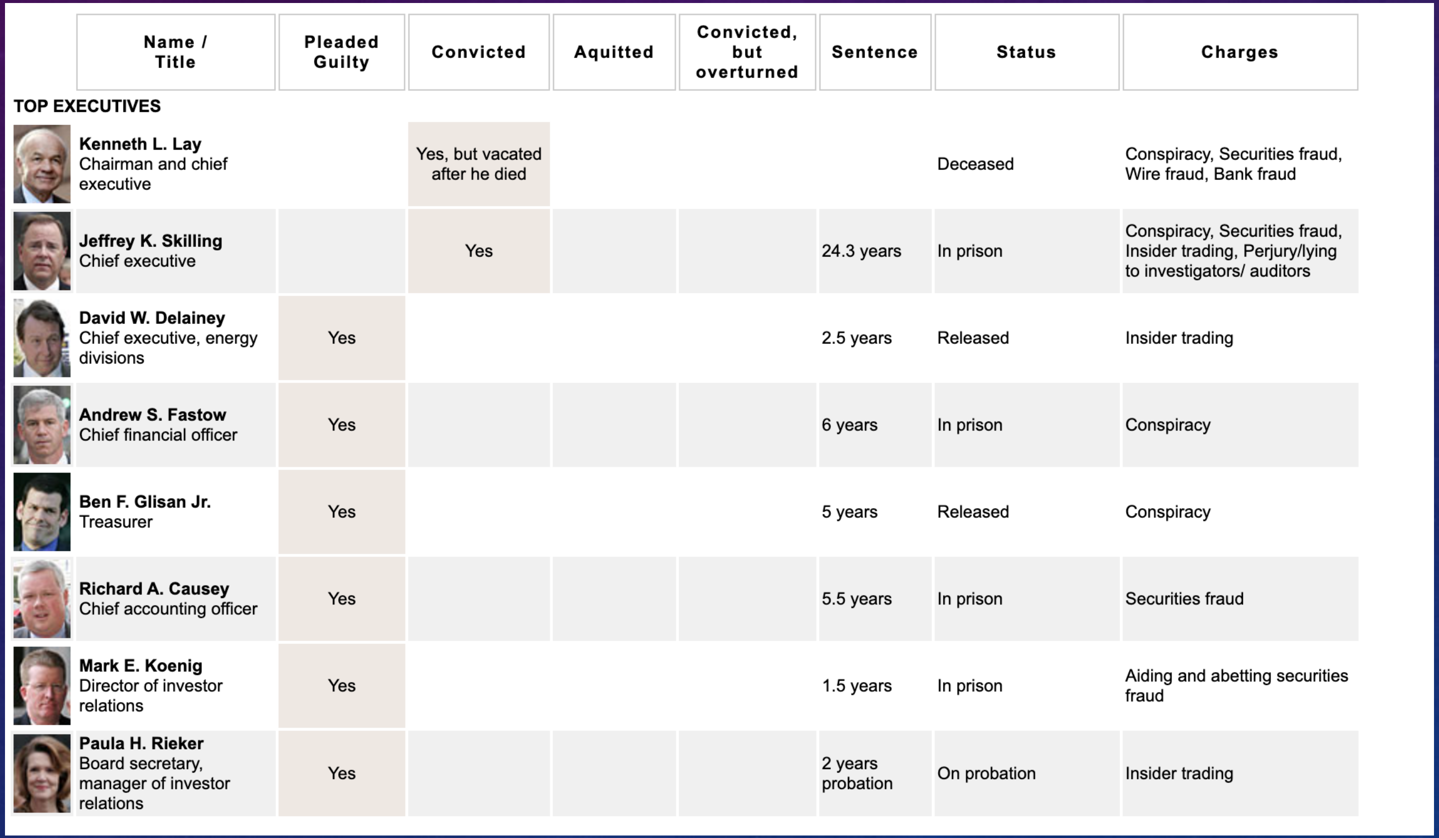
WHAT HAPPENED IN ENRON

In 2000, Enron was one of the largest companies in the United States. By 2002, it had collapsed into bankruptcy due to widespread corporate fraud. In the resulting Federal investigation, there was a significant amount of typically confidential information entered into public record, including tens of thousands of emails and detailed financial data for top executives. The target of the predictions were persons-of-interest (POI's) who were 'individuals who were indicted, reached a settlement, or plea deal with the government, or testified in exchange for prosecution immunity.' Financial compensation data and aggregate email statistics from the Enron Corpus were used as features for prediction.

IDENTIFYING THE NEED FOR INTERVENTION

- Enron's complex financial statements were confusing to shareholders and analysts. In addition, its complex business model and unethical practices required that the company use accounting limitations to misrepresent earnings and modify the balance sheet to indicate favorable performance. Furthermore, some speculative business ventures proved disastrous.
- The combination of these issues later resulted in the bankruptcy of Enron, and most of them were perpetuated by the indirect knowledge or direct actions of Lay, Jeffrey Skilling, Andrew Fastow, and other executives such as Rebecca Mark. Lay served as the chairman of Enron in its last few years, and approved of the actions of Skilling and Fastow, although he did not always inquire about the details.

EXECUTIVES AT ENRON



Name / Title	Pleaded Guilty	Convicted	Aquitted	Convicted, but overturned	Sentence	Status	Charges
TOP EXECUTIVES							
Kenneth L. Lay Chairman and chief executive		Yes, but vacated after he died				Deceased	Conspiracy, Securities fraud, Wire fraud, Bank fraud
Jeffrey K. Skilling Chief executive		Yes			24.3 years	In prison	Conspiracy, Securities fraud, Insider trading, Perjury/lying to investigators/ auditors
David W. Delainey Chief executive, energy divisions	Yes				2.5 years	Released	Insider trading
Andrew S. Fastow Chief financial officer	Yes				6 years	In prison	Conspiracy
Ben F. Glisan Jr. Treasurer	Yes				5 years	Released	Conspiracy
Richard A. Causey Chief accounting officer	Yes				5.5 years	In prison	Securities fraud
Mark E. Koenig Director of investor relations	Yes				1.5 years	In prison	Aiding and abetting securities fraud
Paula H. Rieker Board secretary, manager of investor relations	Yes				2 years probation	On probation	Insider trading

Data from: New York Times Archive

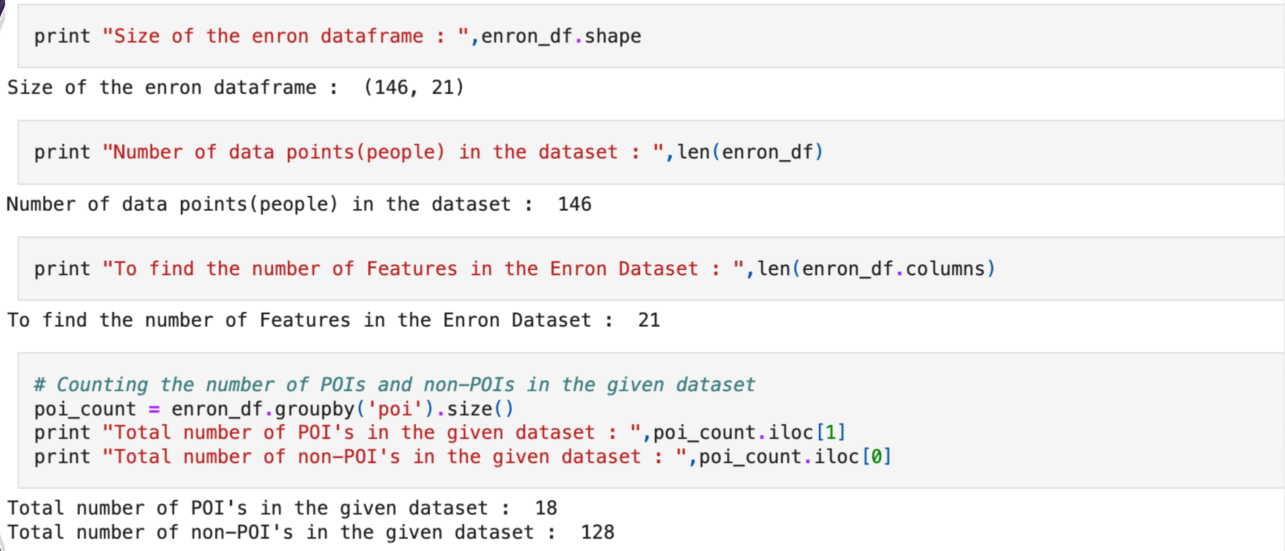
ABOUT THE DATASET

- The dataset contains records of 146 people (thus 146 records of for each feature (including missing values)).
- There are 143 records with 20 features and a binary classification “poi”.
- The 146 records are split in 18 ‘poi’ and 128 ‘non-poi’
- There are considerable amounts of missing values can be observable in all most every feature (for example: the features salary, bonus and to_messages have 51, 64 and 50 missing values, respectively).

DATA PROCESSING

- In the preprocessed dataset, the email and financial information are confined into 21 features of each person investigated. For the preliminary exploration, following list of features selected out of 21 features based on intuition. The number of features will be further reduced in a later stage.
- features_list = ['poi','salary', 'deferral_payments', 'total_payments', 'loan_advances', 'bonus', 'deferred_income', 'total_stock_value','expenses', 'long_term_incentive', 'to_messages', from_poi_to_this_person', 'from_messages', 'from_this_person_to_poi', 'shared_receipt_with_poi']

DATA PROCESSING



```
print "Size of the enron dataframe : ",enron_df.shape
```

```
Size of the enron dataframe :  (146, 21)
```

```
print "Number of data points(people) in the dataset : ",len(enron_df)
```

```
Number of data points(people) in the dataset :  146
```

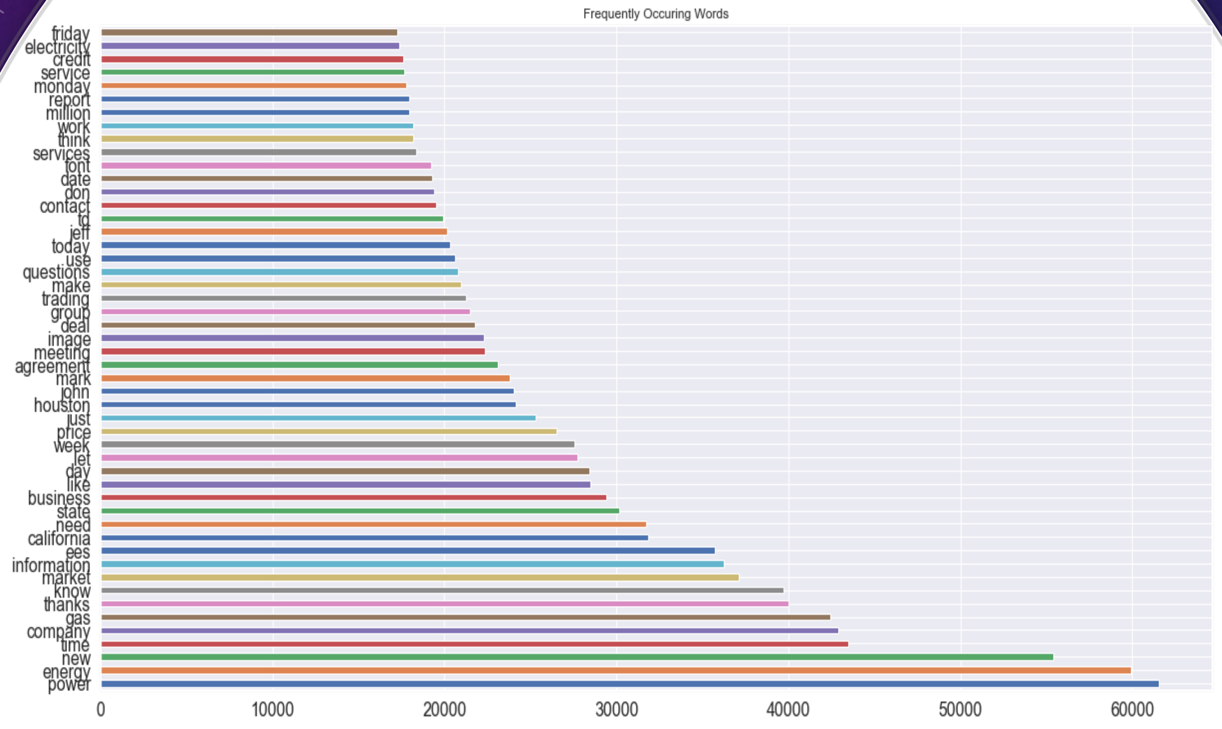
```
print "To find the number of Features in the Enron Dataset : ",len(enron_df.columns)
```

```
To find the number of Features in the Enron Dataset :  21
```

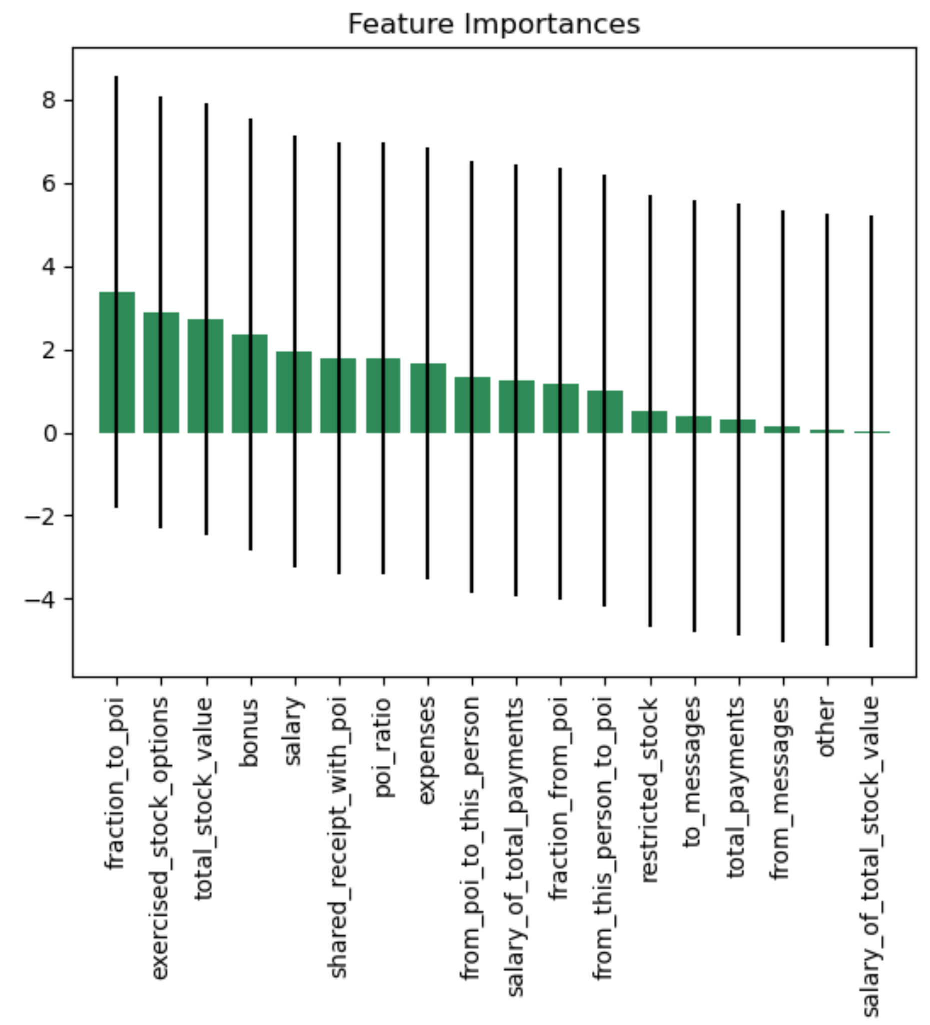
```
# Counting the number of POIs and non-POIs in the given dataset
poi_count = enron_df.groupby('poi').size()
print "Total number of POI's in the given dataset : ",poi_count.iloc[1]
print "Total number of non-POI's in the given dataset : ",poi_count.iloc[0]
```

```
Total number of POI's in the given dataset :  18
Total number of non-POI's in the given dataset :  128
```

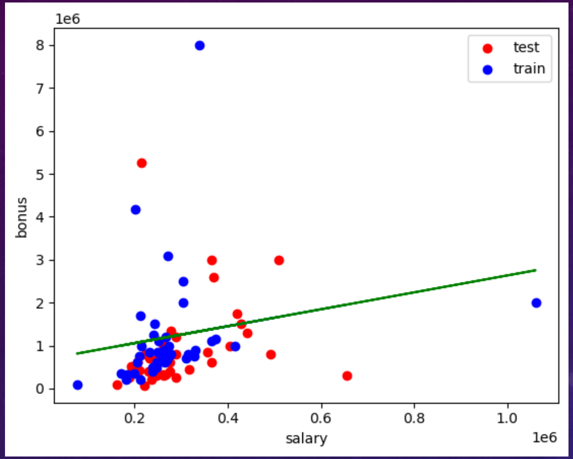
KEY WORD ANALYSIS

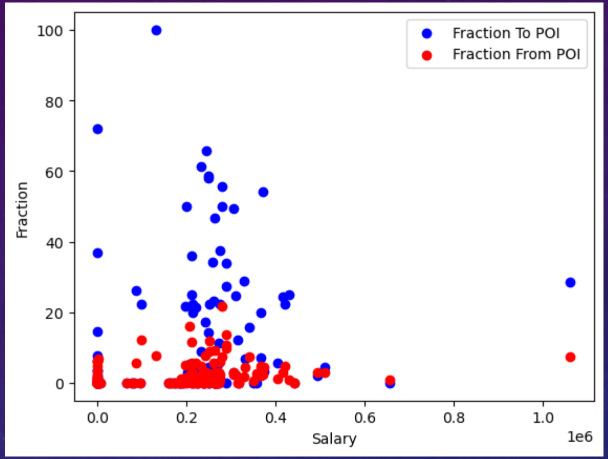


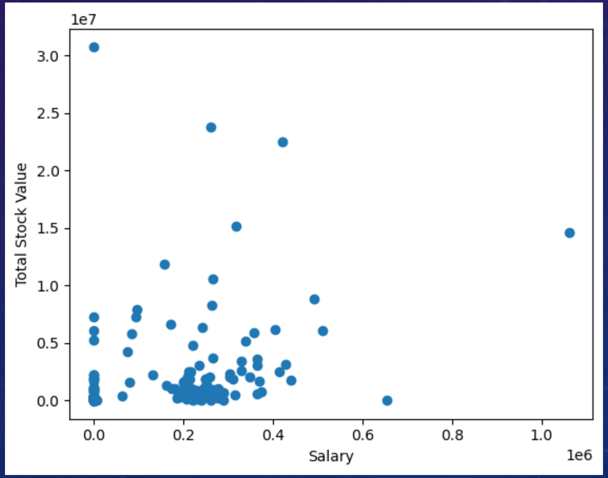
FEATURE IMPORTANCE

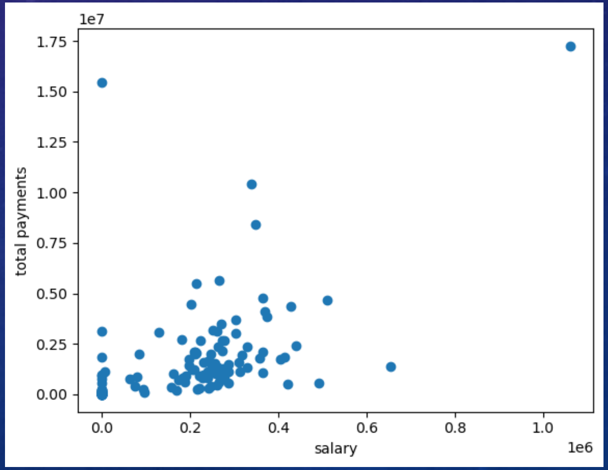


FEATURE COMPARISONS

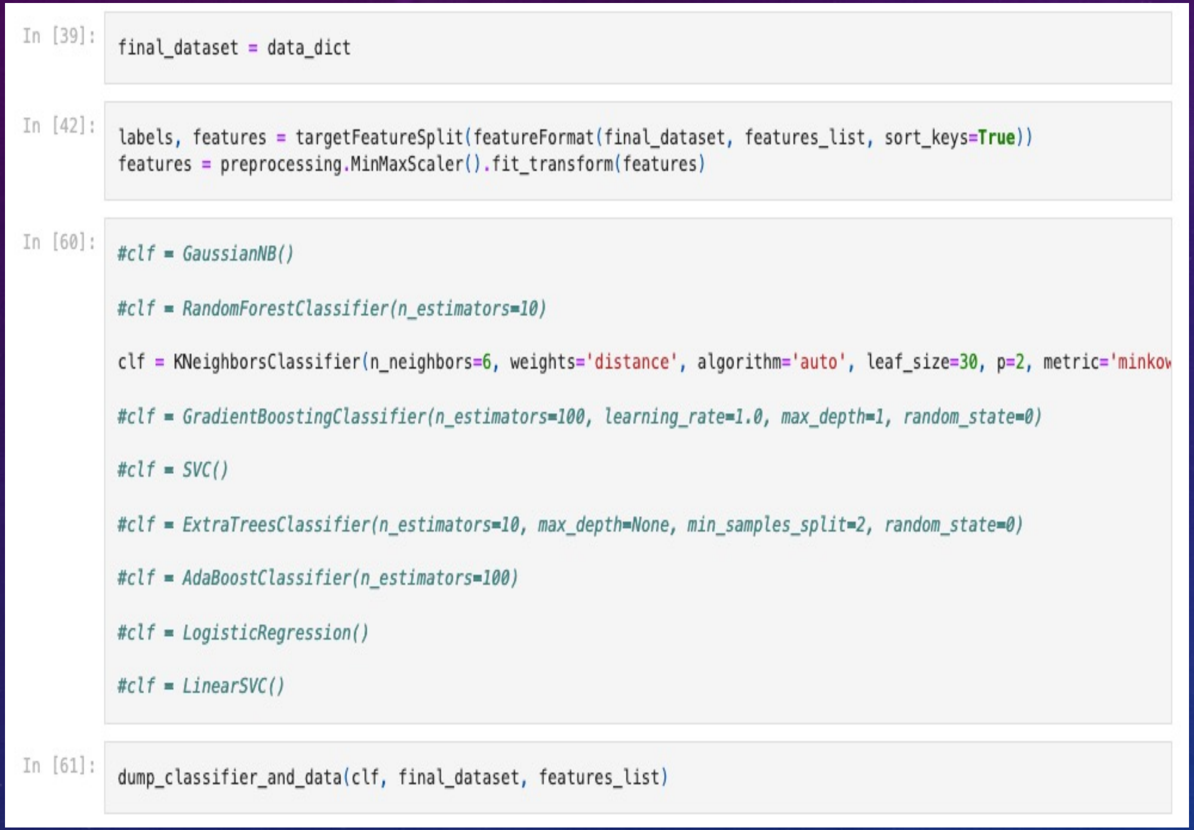




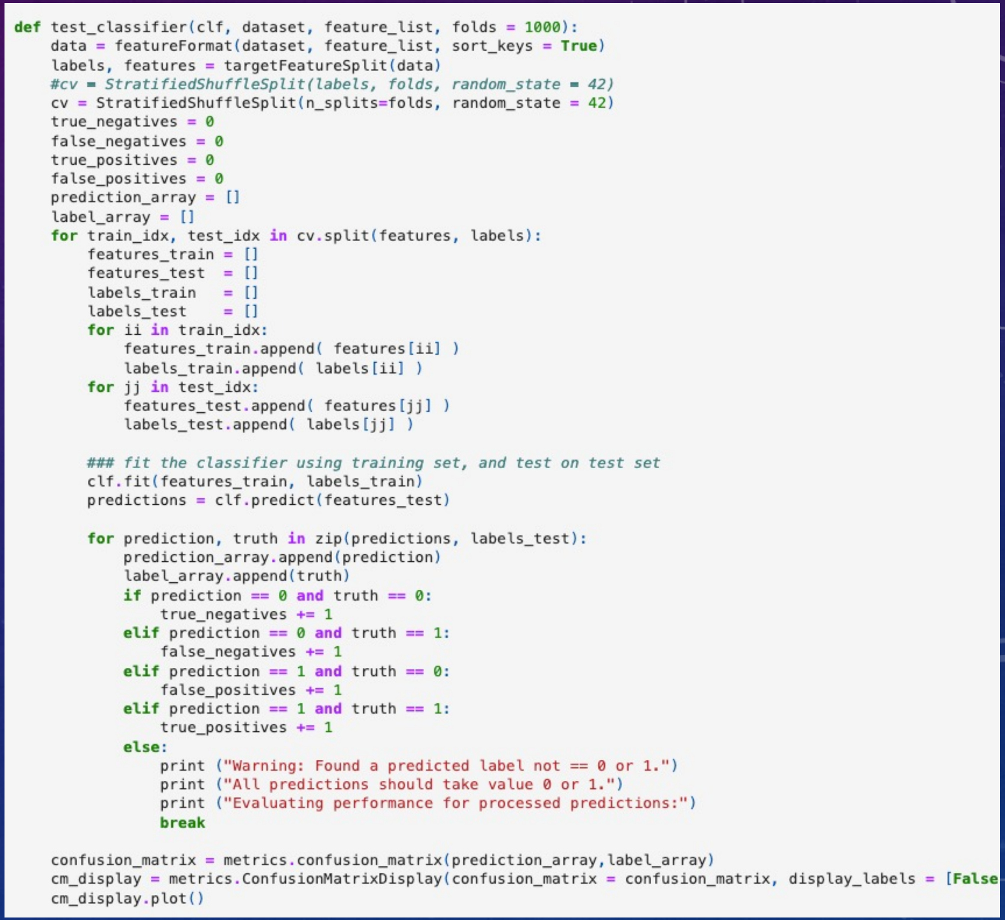




CODE



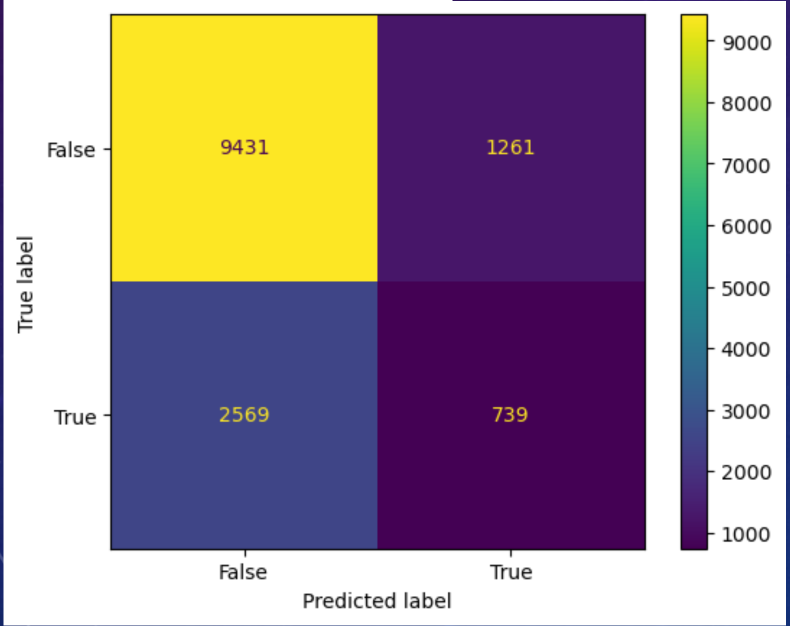
```
In [39]: final_dataset = data_dict

In [42]: labels, features = targetFeatureSplit(featureFormat(final_dataset, features_list, sort_keys=True))
         features = preprocessing.MinMaxScaler().fit_transform(features)

In [60]: #clf = GaussianNB()

         #clf = RandomForestClassifier(n_estimators=10)

         clf = KNeighborsClassifier(n_neighbors=6, weights='distance', algorithm='auto', leaf_size=30, p=2, metric='minkow

         #clf = GradientBoostingClassifier(n_estimators=100, learning_rate=1.0, max_depth=1, random_state=0)

         #clf = SVC()

         #clf = ExtraTreesClassifier(n_estimators=10, max_depth=None, min_samples_split=2, random_state=0)

         #clf = AdaBoostClassifier(n_estimators=100)

         #clf = LogisticRegression()

         #clf = LinearSVC()

In [61]: dump_classifier_and_data(clf, final_dataset, features_list)
```



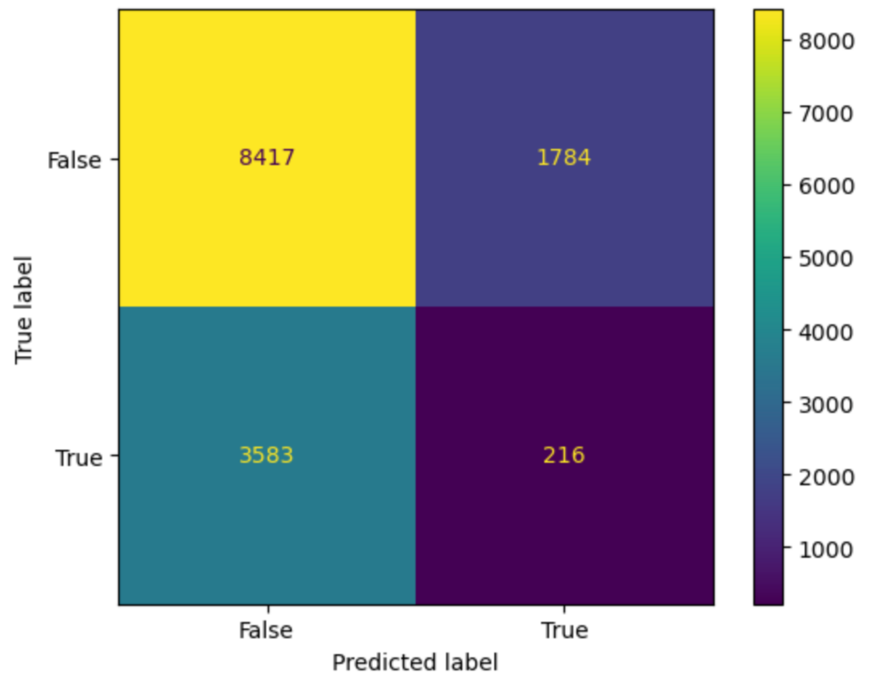
```
def test_classifier(clf, dataset, feature_list, folds = 1000):
    data = featureFormat(dataset, feature_list, sort_keys = True)
    labels, features = targetFeatureSplit(data)
    #cv = StratifiedShuffleSplit(labels, folds, random_state = 42)
    cv = StratifiedShuffleSplit(n_splits=folds, random_state = 42)
    true_negatives = 0
    false_negatives = 0
    true_positives = 0
    false_positives = 0
    prediction_array = []
    label_array = []
    for train_idx, test_idx in cv.split(features, labels):
        features_train = []
        features_test  = []
        labels_train   = []
        labels_test    = []
        for ii in train_idx:
            features_train.append( features[ii] )
            labels_train.append( labels[ii] )
        for jj in test_idx:
            features_test.append( features[jj] )
            labels_test.append( labels[jj] )

        ### fit the classifier using training set, and test on test set
        clf.fit(features_train, labels_train)
        predictions = clf.predict(features_test)

        for prediction, truth in zip(predictions, labels_test):
            prediction_array.append(prediction)
            label_array.append(truth)
            if prediction == 0 and truth == 0:
                true_negatives += 1
            elif prediction == 0 and truth == 1:
                false_negatives += 1
            elif prediction == 1 and truth == 0:
                false_positives += 1
            elif prediction == 1 and truth == 1:
                true_positives += 1
            else:
                print ("Warning: Found a predicted label not == 0 or 1.")
                print ("All predictions should take value 0 or 1.")
                print ("Evaluating performance for processed predictions:")
                break

    confusion_matrix = metrics.confusion_matrix(prediction_array,label_array)
    cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = confusion_matrix, display_labels = [False
    cm_display.plot()
```

OUTPUT

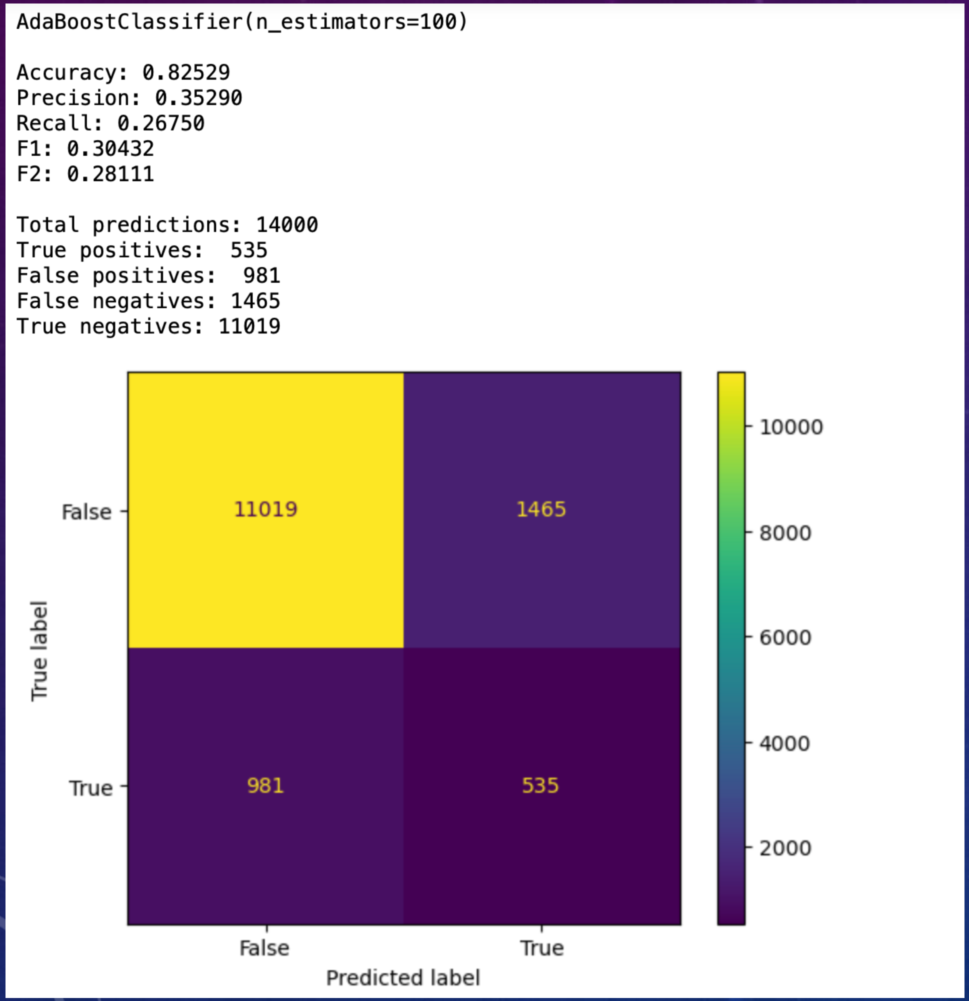
```
LinearSVC()

Accuracy: 0.72643
Precision: 0.22340
Recall: 0.36950
F1: 0.27845
F2: 0.32676

Total predictions: 14000
True positives:  739
False positives: 2569
False negatives: 1261
True negatives: 9431
```



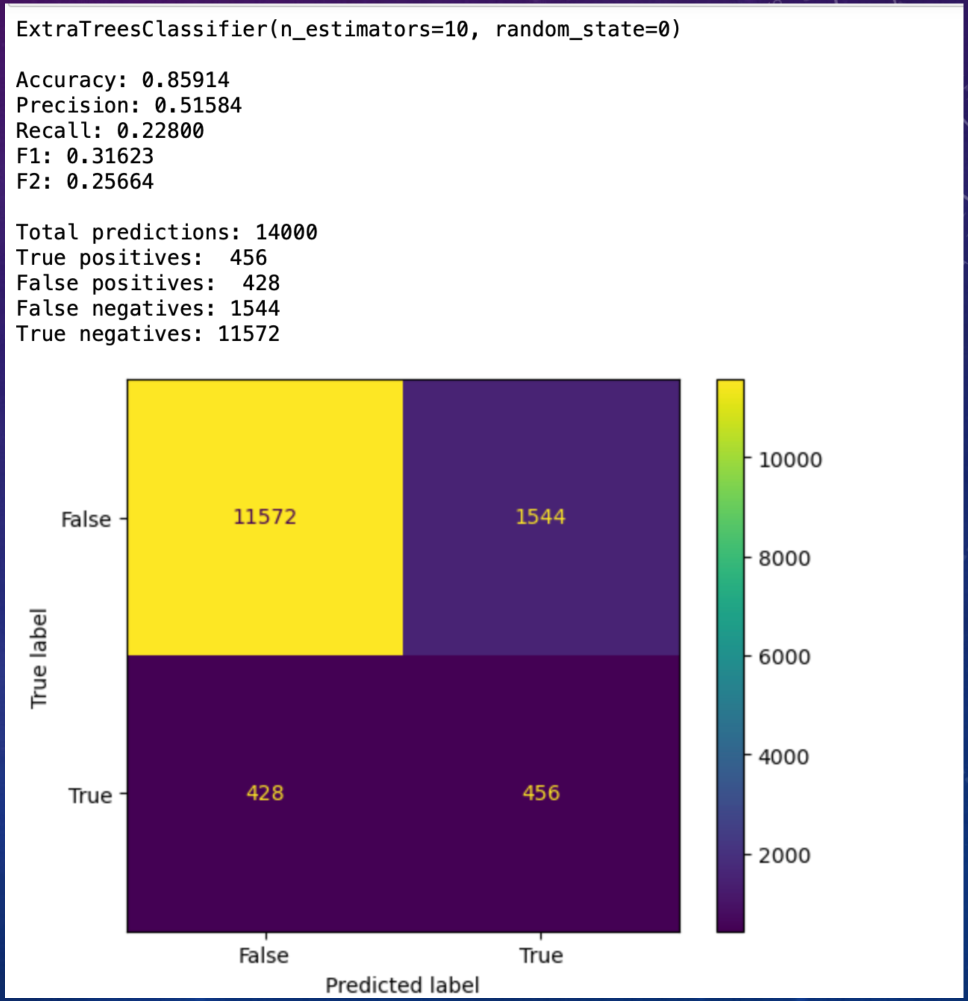
```
LogisticRegression()

Accuracy: 0.61664
Precision: 0.05686
Recall: 0.10800
F1: 0.07450
F2: 0.09153

Total predictions: 14000
True positives:  216
False positives: 3583
False negatives: 1784
True negatives: 8417
```



OUTPUT

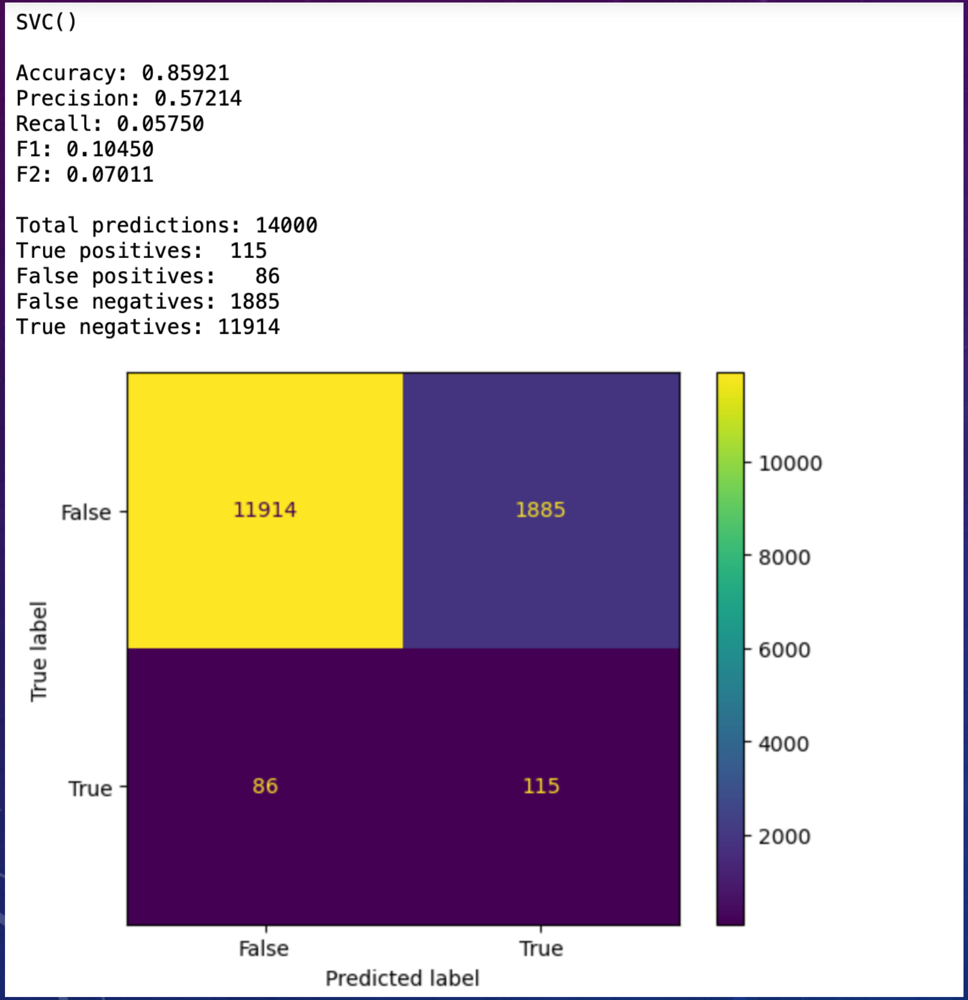


```
AdaBoostClassifier(n_estimators=100)

Accuracy: 0.82529
Precision: 0.35290
Recall: 0.26750
F1: 0.30432
F2: 0.28111

Total predictions: 14000
True positives:  535
False positives:  981
False negatives: 1465
True negatives: 11019
```

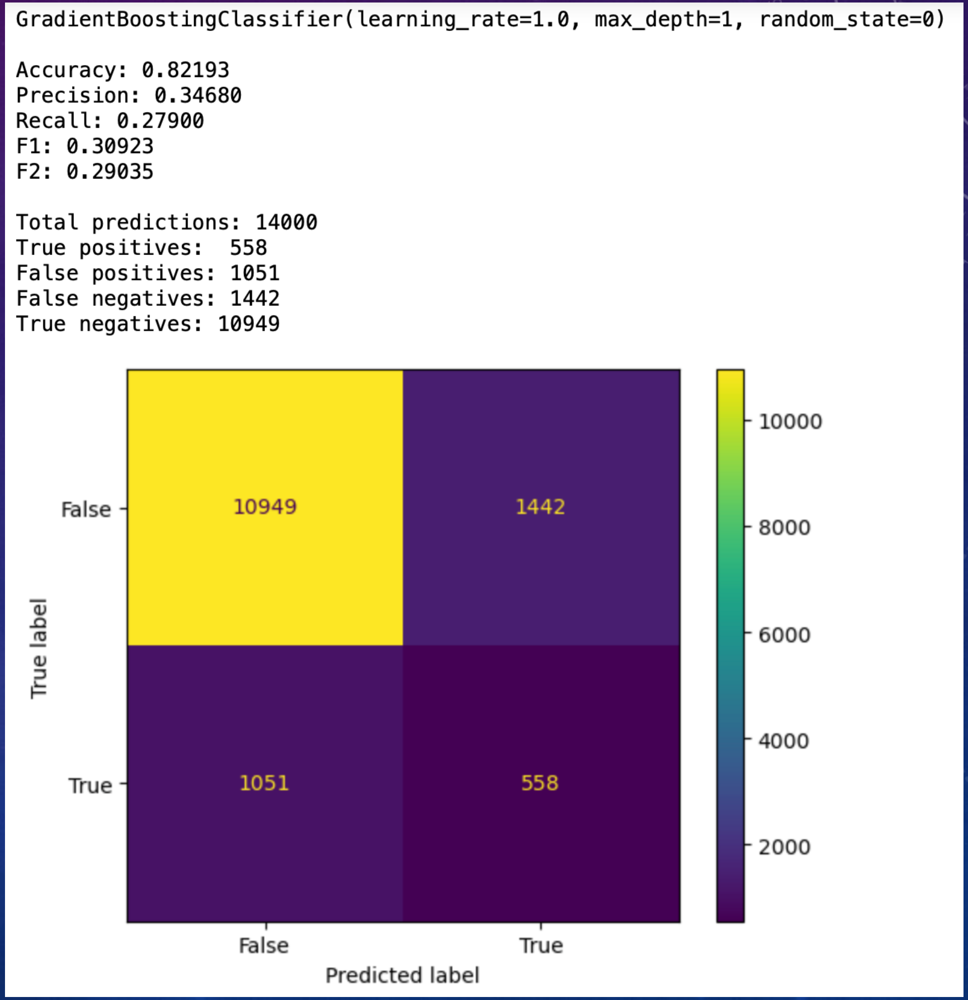


```
ExtraTreesClassifier(n_estimators=10, random_state=0)

Accuracy: 0.85914
Precision: 0.51584
Recall: 0.22800
F1: 0.31623
F2: 0.25664

Total predictions: 14000
True positives:  456
False positives:  428
False negatives: 1544
True negatives: 11572
```

OUTPUT

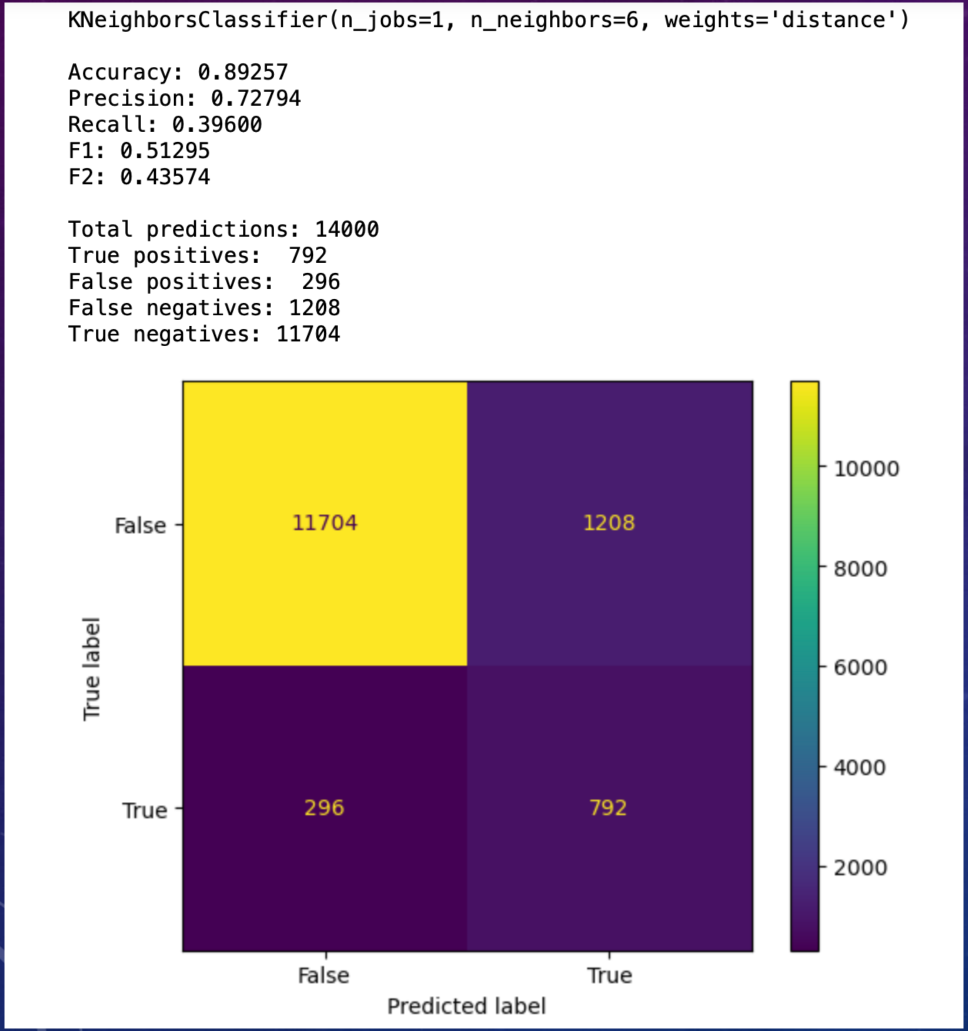


```
SVC()

Accuracy: 0.85921
Precision: 0.57214
Recall: 0.05750
F1: 0.10450
F2: 0.07011

Total predictions: 14000
True positives:  115
False positives:   86
False negatives: 1885
True negatives: 11914
```

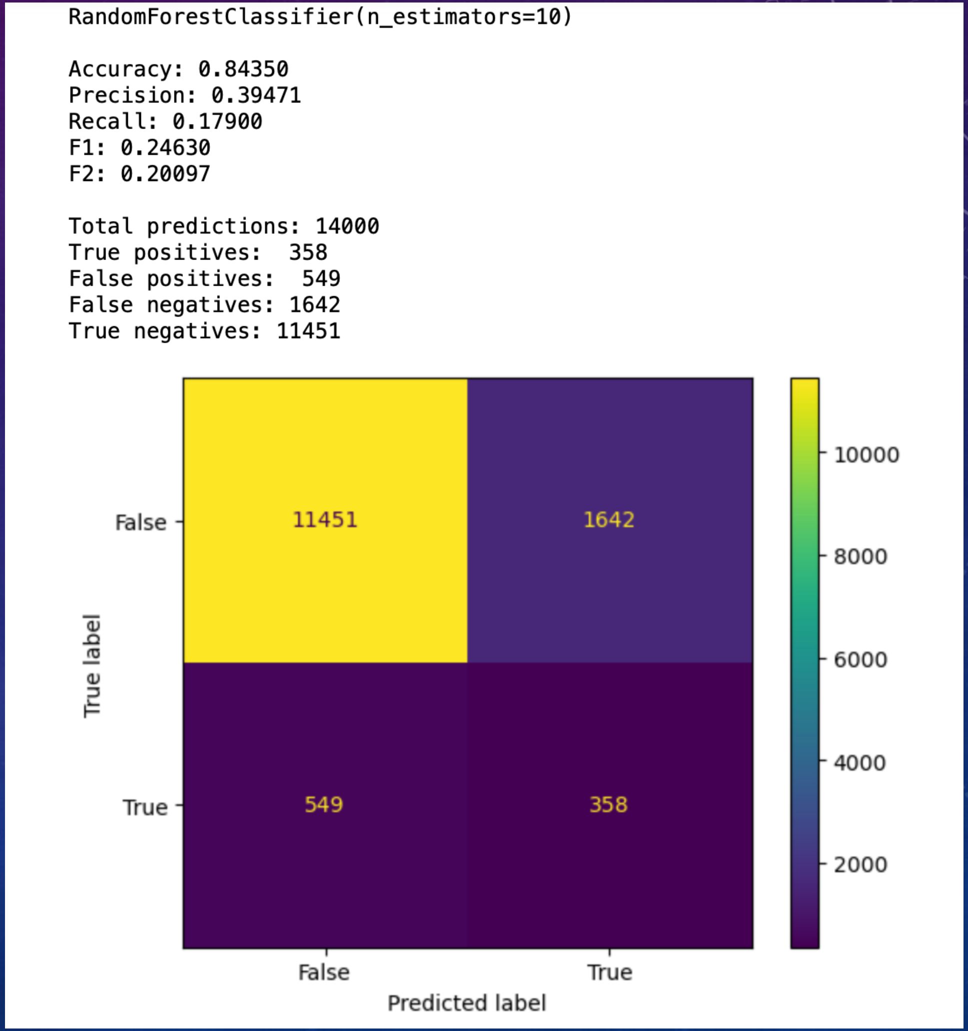


```
GradientBoostingClassifier(learning_rate=1.0, max_depth=1, random_state=0)

Accuracy: 0.82193
Precision: 0.34680
Recall: 0.27900
F1: 0.30923
F2: 0.29035

Total predictions: 14000
True positives:  558
False positives: 1051
False negatives: 1442
True negatives: 10949
```

OUTPUT



```
KNeighborsClassifier(n_jobs=1, n_neighbors=6, weights='distance')

Accuracy: 0.89257
Precision: 0.72794
Recall: 0.39600
F1: 0.51295
F2: 0.43574

Total predictions: 14000
True positives:  792
False positives:  296
False negatives: 1208
True negatives: 11704
```



```
RandomForestClassifier(n_estimators=10)

Accuracy: 0.84350
Precision: 0.39471
Recall: 0.17900
F1: 0.24630
F2: 0.20097

Total predictions: 14000
True positives:  358
False positives:  549
False negatives: 1642
True negatives: 11451
```

GITHUB

https://github.com/Ashay1301/Enron-POI.git