CSE 523: Machine Learning

Group 17 - Hardly Humans

Weekly Project Report - 10

Quora Insincere Questions Classification

Name	**Enrolment Number**
Malav Doshi	AU1940017
Parth Shah	AU1940065
Sanya Zaveri	AU1920064
Mihir Pathak	AU1920138

1) Tasks Performed in the week.

Implemented SVM.

Implemented LDA.

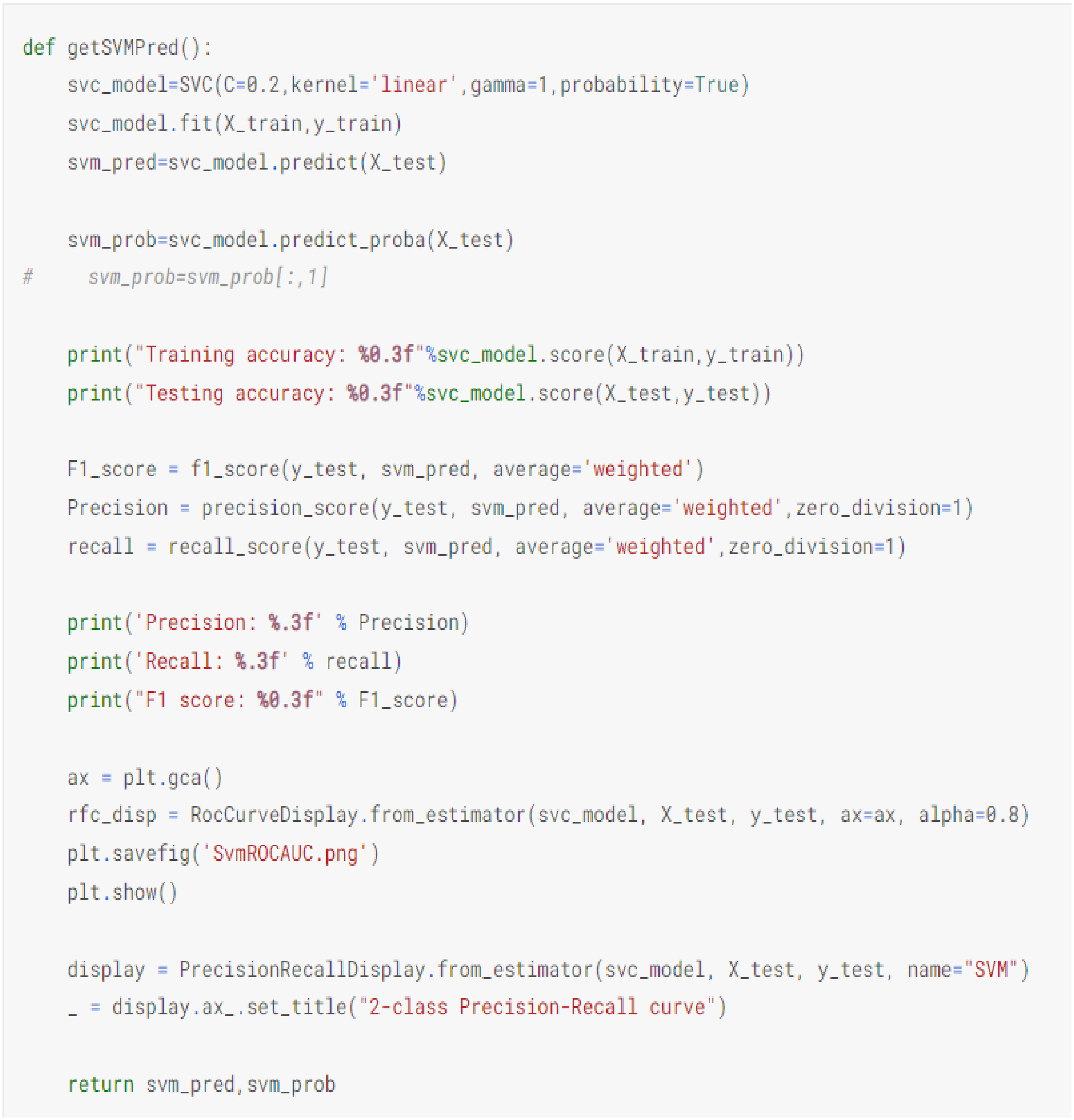
Implemented Random Forest.

2) Outcomes of the tasks performed.

- Models that were fitted to SVM and Random forests respectively.
    1) Model fitting for SVM

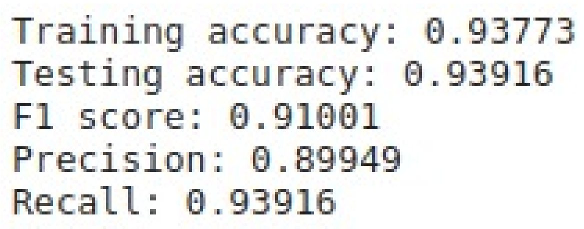
Using SVC from sklearn.svm with parameters - C=0.2,kernel='linear',gamma=1

We train the mode

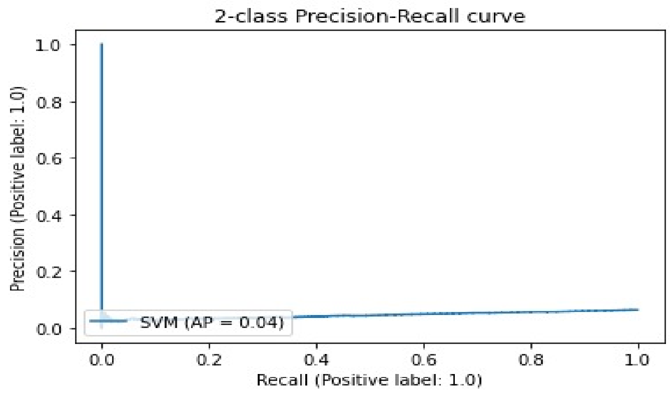


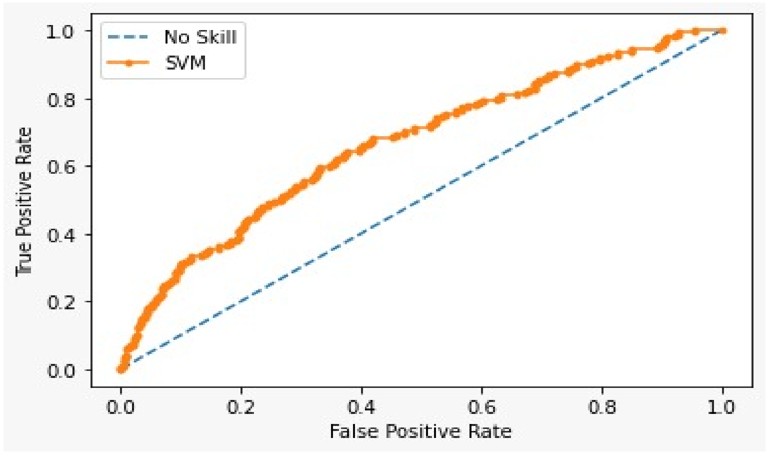
```python
def getSVMPred():
    svc_model=SVC(C=0.2,kernel='linear',gamma=1,probability=True)
    svc_model.fit(X_train,y_train)
    svm_pred=svc_model.predict(X_test)

    svm_prob=svc_model.predict_proba(X_test)
#     svm_prob=svm_prob[:,1]

    print("Training accuracy: %0.3f"%svc_model.score(X_train,y_train))
    print("Testing accuracy: %0.3f"%svc_model.score(X_test,y_test))

    F1_score = f1_score(y_test, svm_pred, average='weighted')
    Precision = precision_score(y_test, svm_pred, average='weighted',zero_division=1)
    recall = recall_score(y_test, svm_pred, average='weighted',zero_division=1)

    print('Precision: %.3f' % Precision)
    print('Recall: %.3f' % recall)
    print("F1 score: %0.3f" % F1_score)

    ax = plt.gca()
    rfc_disp = RocCurveDisplay.from_estimator(svc_model, X_test, y_test, ax=ax, alpha=0.8)
    plt.savefig('SvmROCAUC.png')
    plt.show()

    display = PrecisionRecallDisplay.from_estimator(svc_model, X_test, y_test, name="SVM")
    _ = display.ax_.set_title("2-class Precision-Recall curve")

    return svm_pred,svm_prob
```

Results:



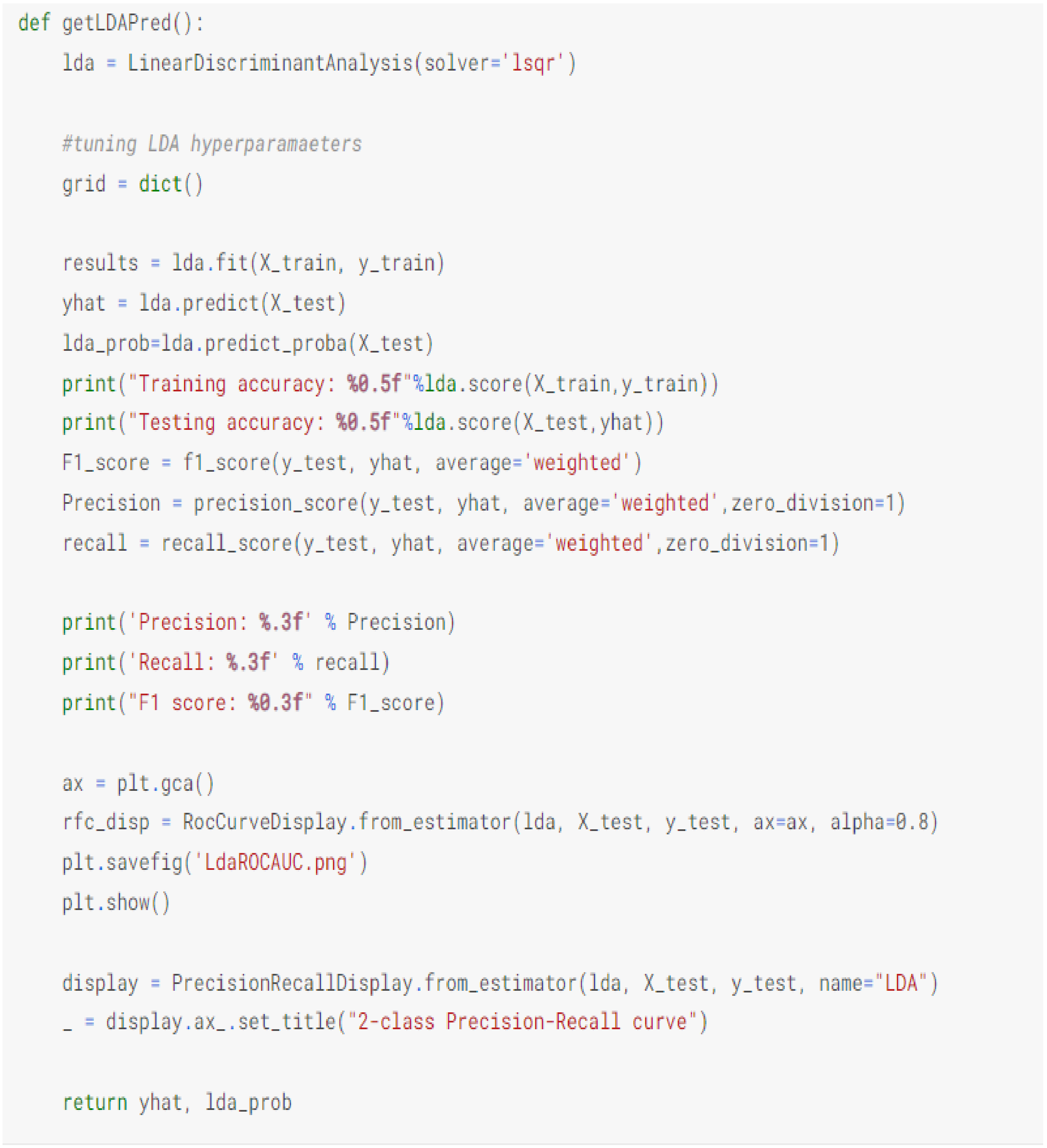
```
Training accuracy: 0.93773
Testing accuracy: 0.93916
F1 score: 0.91001
Precision: 0.89949
Recall: 0.93916
```


2-class Precision-Recall curve
Precision (Positive label: 1.0)
Recall (Positive label: 1.0)
SVM (AP = 0.04)
0.0
0.2
0.4
0.6
0.8
1.0


No Skill
SVM
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0

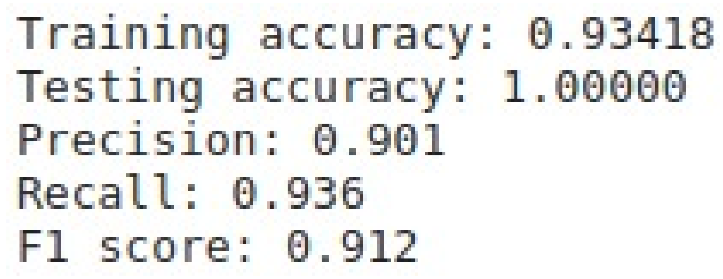
2) Model fitting for LDA

Imported Linear Discriminant Analysis model from sklearn.LinearDiscriminantAnalysis

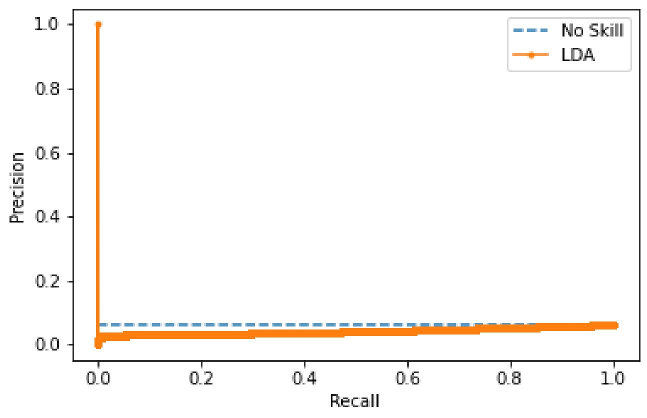


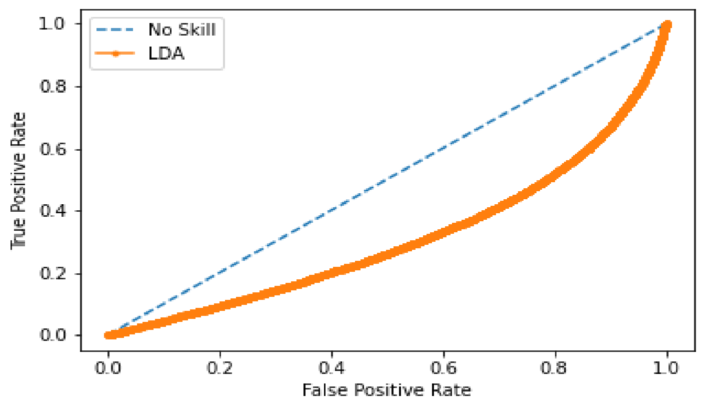
```python
def getLDAPred():
    lda = LinearDiscriminantAnalysis(solver='lsqr')

    #tuning LDA hyperparamaeters
    grid = dict()

    results = lda.fit(X_train, y_train)
    yhat = lda.predict(X_test)
    lda_prob=lda.predict_proba(X_test)
    print("Training accuracy: %0.5f"%lda.score(X_train,y_train))
    print("Testing accuracy: %0.5f"%lda.score(X_test,yhat))
    F1_score = f1_score(y_test, yhat, average='weighted')
    Precision = precision_score(y_test, yhat, average='weighted',zero_division=1)
    recall = recall_score(y_test, yhat, average='weighted',zero_division=1)

    print('Precision: %.3f' % Precision)
    print('Recall: %.3f' % recall)
    print("F1 score: %0.3f" % F1_score)

    ax = plt.gca()
    rfc_disp = RocCurveDisplay.from_estimator(lda, X_test, y_test, ax=ax, alpha=0.8)
    plt.savefig('LdaROCAUC.png')
    plt.show()

    display = PrecisionRecallDisplay.from_estimator(lda, X_test, y_test, name="LDA")
    _ = display.ax_.set_title("2-class Precision-Recall curve")

    return yhat, lda_prob
```

Results:



```
Training accuracy: 0.93418
Testing accuracy: 1.00000
Precision: 0.901
Recall: 0.936
F1 score: 0.912
```





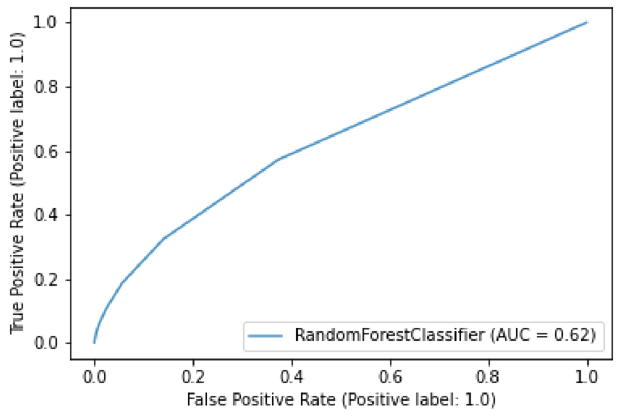
3) Model fitting for Random forest

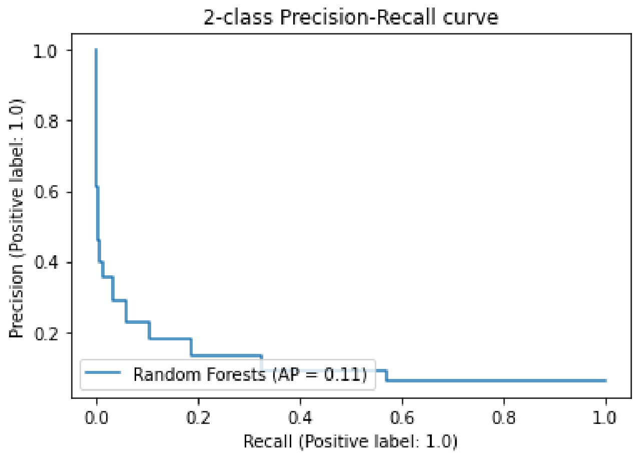
Using RandomForestClassifier from sklearn.ensemble.

```
def getRandomForest():
    RF_model = RandomForestClassifier(n_estimators=10, random_state=42)
    RF_model.fit(X_train,y_train)
    RF_preds = RF_model.predict(X_test)
    RF_prob=RF_model.predict_proba(X_test)
    print("Training accuracy: %0.3f"%RF_model.score(X_train,y_train))
    print("Testing accuracy: %0.3f"%RF_model.score(X_test,y_test))

    F1_score1 = f1_score(y_test, RF_preds, average='weighted')
    Precision1 = precision_score(y_test, RF_preds, average='weighted',zero_division=1)
    recall1 = recall_score(y_test, RF_preds, average='weighted',zero_division=1)

    print('Precision: %.3f' % Precision1)
    print('Recall: %.3f' % recall1)
    print("F1 score: %0.3f" % F1_score1)

    ax = plt.gca()
    rfc_disp = RocCurveDisplay.from_estimator(RF_model, X_test, y_test, ax=ax, alpha=0.8)
    plt.savefig('RandForestROCAUC.png')
    plt.show()

    display = PrecisionRecallDisplay.from_estimator(RF_model, X_test, y_test, name="Random F
orests")
    _ = display.ax_.set_title("2-class Precision-Recall curve")

    return RF_preds,RF_prob
```

Results:

```
Training accuracy: 0.988
Testing accuracy: 0.938
Precision: 0.905
Recall: 0.938
F1 score: 0.912
```


True Positive Rate (Positive label: 1.0)
False Positive Rate (Positive label: 1.0)
RandomForestClassifier (AUC = 0.62)


2-class Precision-Recall curve
Precision (Positive label: 1.0)
Recall (Positive label: 1.0)
Random Forests (AP = 0.11)