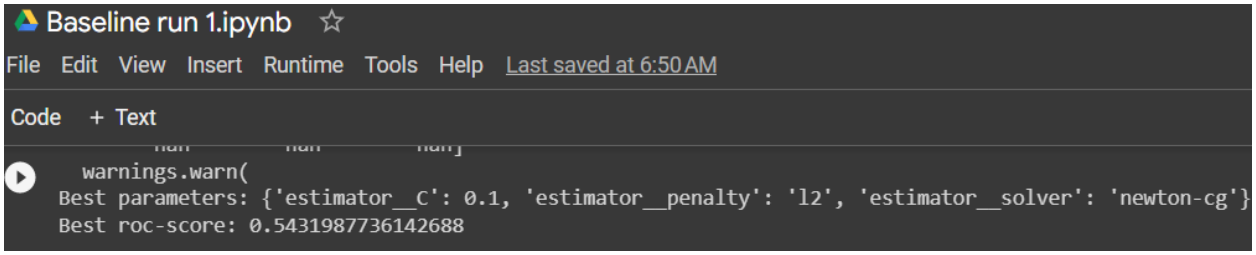
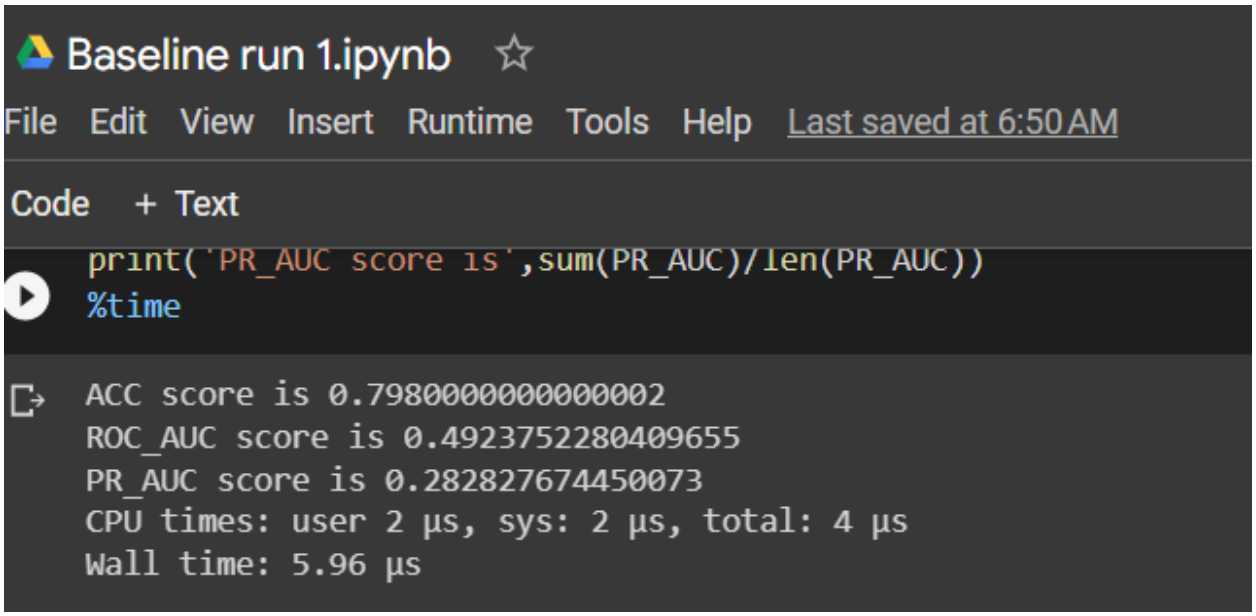
In this file we present the results screenshots

1. Baseline

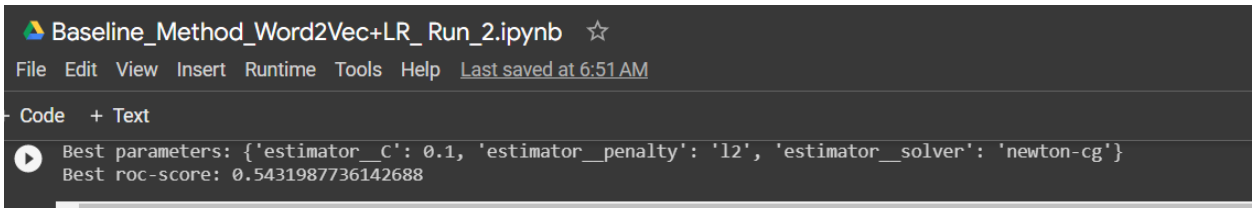
1a. Parameter Tuning for Baseline Run1



1b. Test results for Baseline Run 1



1c. Parameter Tuning for Baseline Run 2



1d. Test Results for Baseline Run 2

Baseline_Method_Word2Vec+LR_ Run_2.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 6:51 AM

Code + Text

```
[ ]     ACC.append(acc)
        ROC_AUC.append(auc_roc)
    print('ACC score is',sum(ACC)/len(ACC))
    print('ROC_AUC score is',sum(ROC_AUC)/len(ROC_AUC))
    print('PR_AUC score is',sum(PR_AUC)/len(PR_AUC))
    %time
```

```
ACC score is 0.7980000000000002
ROC_AUC score is 0.4923752280409655
PR_AUC score is 0.282827674450073
CPU times: user 2 µs, sys: 1e+03 ns, total: 3 µs
Wall time: 5.48 µs
```

2. Word2 Vec+ KNN

2a. Parameter Tuning for Word2 Vec+ KNN Run1

Word2Vec+_KNN_Run_1.ipynb ☆

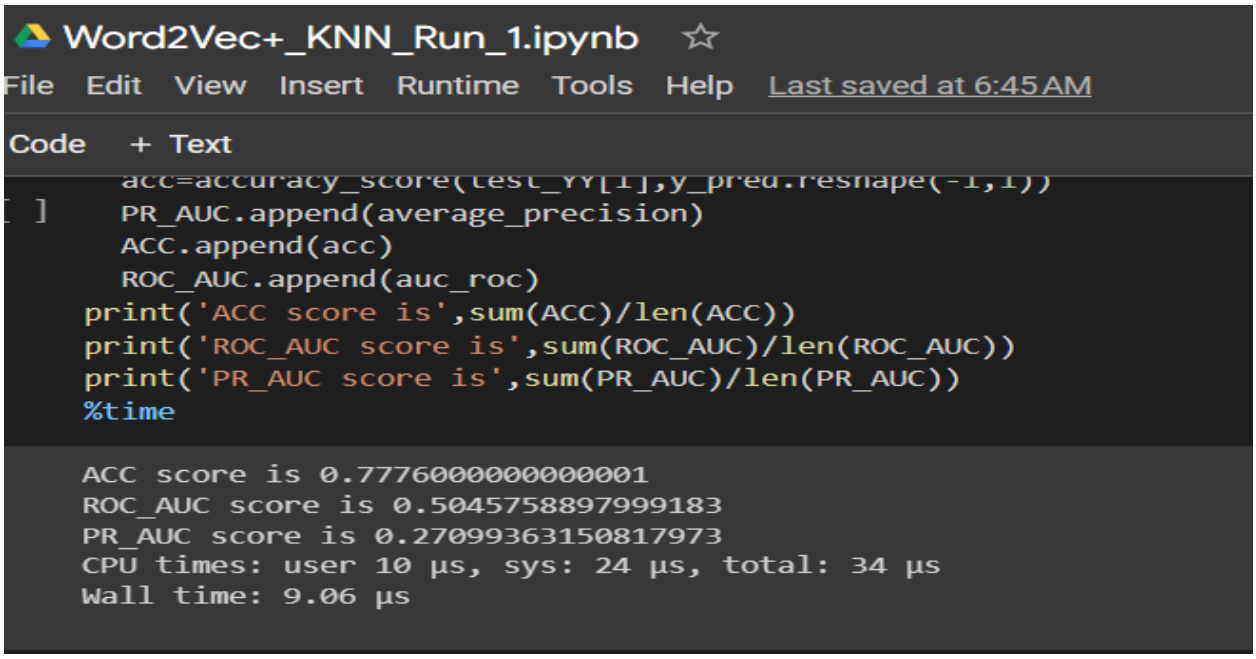
ile Edit View Insert Runtime Tools Help Last saved at 6:45 AM

Code + Text

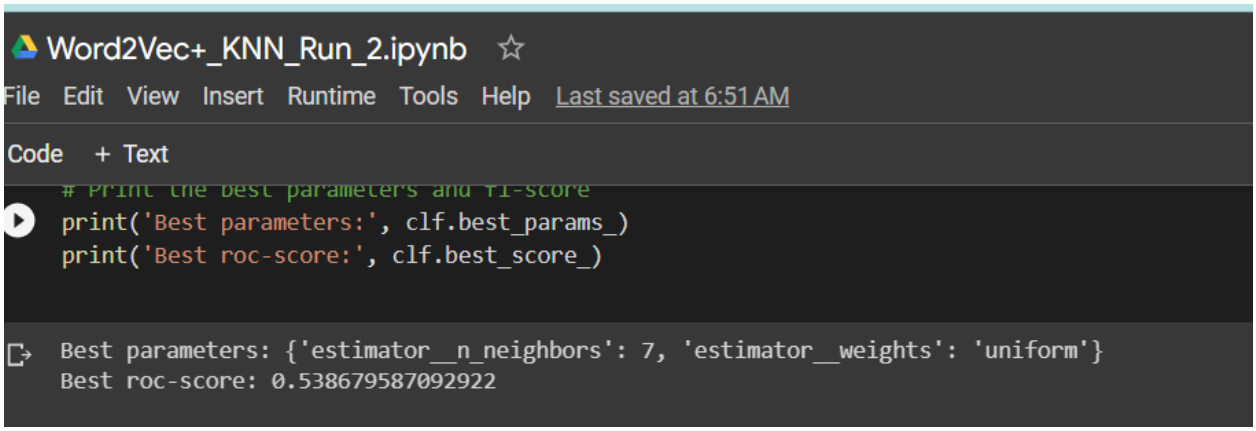
```
]     'estimator__n_neighbors': [3, 5, 7],
      'estimator__weights': ['uniform', 'distance']
  }

  # Define the GridSearchCV object with f1-score as the scoring metric
  clf = GridSearchCV(estimator, parameters, scoring='roc_auc')

  # Fit the GridSearchCV object to the training data
  clf.fit(train_X, YY)

  # Print the best parameters and f1-score
  print('Best parameters:', clf.best_params_)
  print('Best roc-score:', clf.best_score_)
```

```
Best parameters: {'estimator__n_neighbors': 7, 'estimator__weights': 'uniform'}
Best roc-score: 0.538679587092922
```

2b. Test data results for Word2 Vec+ KNN run 1



2c. Parameter Tuning for Word2 Vec+ KNN run 2



2d. Test data results for Word2 Vec+ KNN run 2

Word2Vec+_KNN_Run_2.ipynb

File Edit View Insert Runtime Tools Help Last saved at 6:51 AM

Code + Text

```
        average_precision=0 #setting zero if testing lables doesnot
      acc=accuracy_score(test_YY[i],y_pred.reshape(-1,1))
      PR_AUC.append(average_precision)
      ACC.append(acc)
      ROC_AUC.append(auc_roc)
    print('ACC score is',sum(ACC)/len(ACC))
    print('ROC_AUC score is',sum(ROC_AUC)/len(ROC_AUC))
    print('PR_AUC score is',sum(PR_AUC)/len(PR_AUC))
    %time
```

```
ACC score is 0.7776000000000001
ROC_AUC score is 0.5045758897999183
PR_AUC score is 0.27099363150817973
CPU times: user 10 µs, sys: 37 µs, total: 47 µs
Wall time: 7.87 µs
```

3. Word 2 vec +CNN+LR

3a. Parameter Tuning for Word 2 vec +CNN+LR

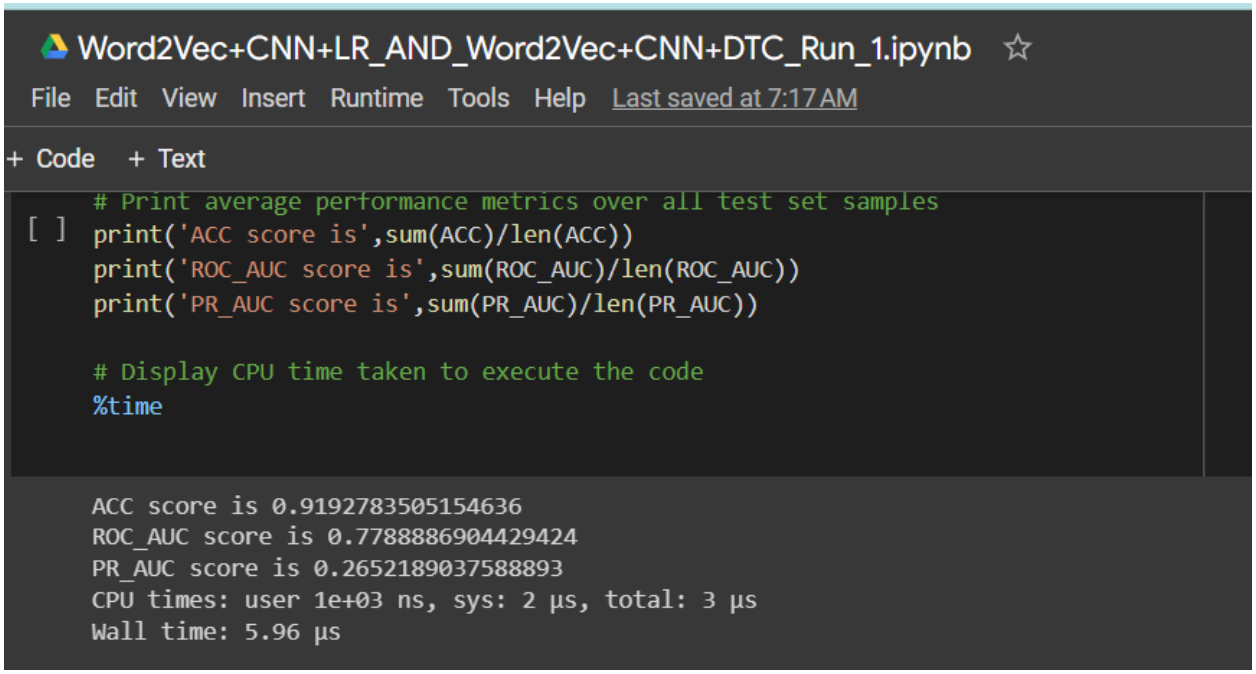
Word2Vec+CNN+LR_AND_Word2Vec+CNN+DTC_Run_1.ipynb

File Edit View Insert Runtime Tools Help Last saved at 7:17 AM

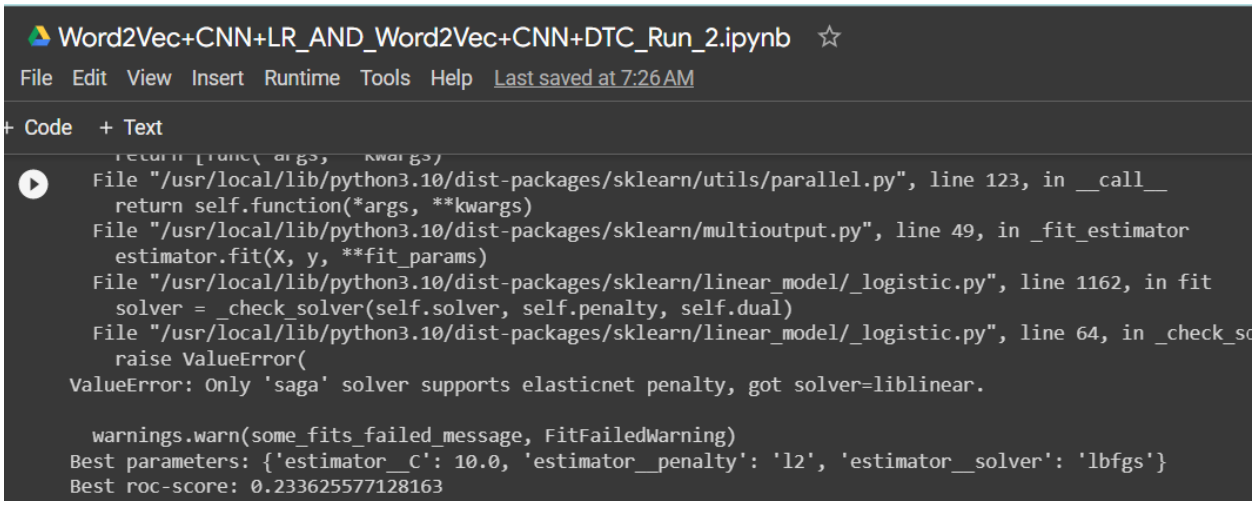
+ Code + Text

```
    solver = _check_solver(self.solver, self.penalty, self.dual)
  File "/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py", line 64, in _check_solver
    raise ValueError(
ValueError: Only 'saga' solver supports elasticnet penalty, got solver=liblinear.

  warnings.warn(some_fits_failed_message, FitFailedWarning)
Best parameters: {'estimator__C': 10.0, 'estimator__penalty': 'l2', 'estimator__solver': 'lbfgs'}
Best roc-score: 0.2474436753967851
```

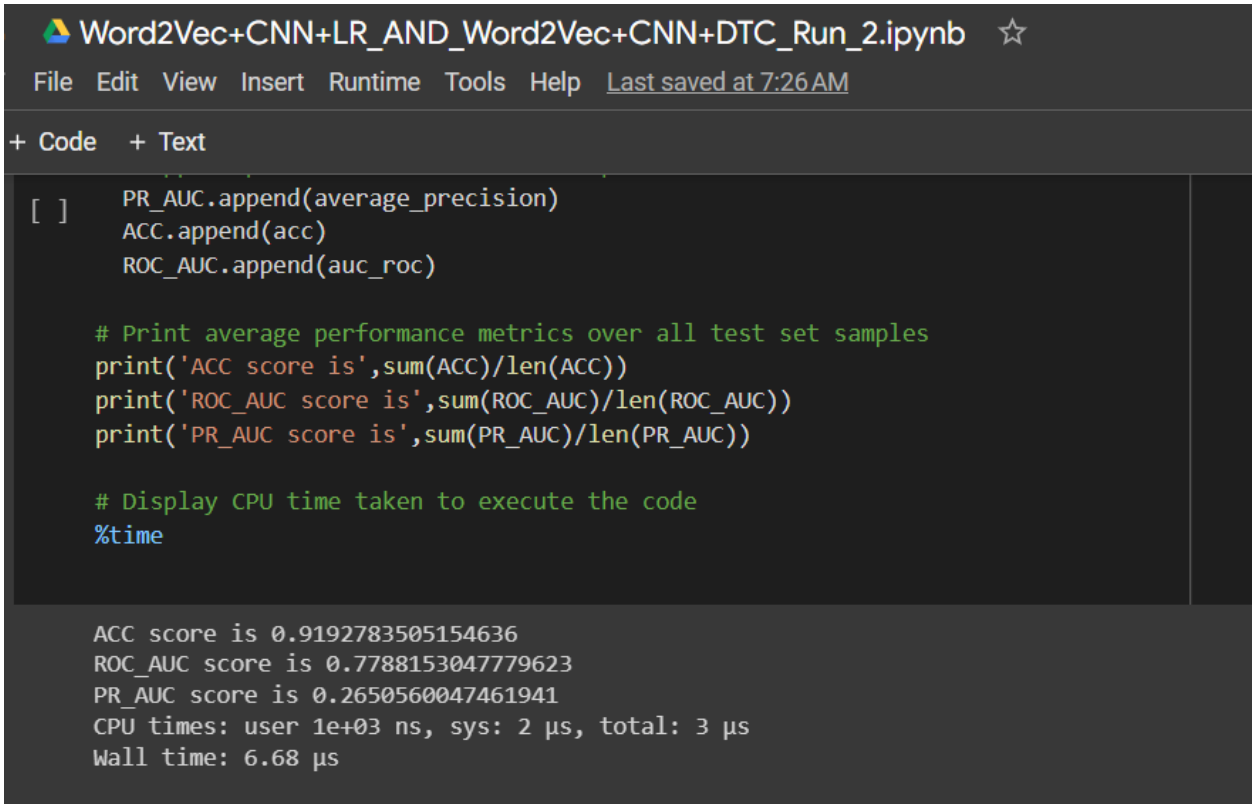
3b. Test data results for Word 2 vec +CNN+ LR run 1



3c. Parameter Tuning for Word 2 vec + CNN+ LR run 2

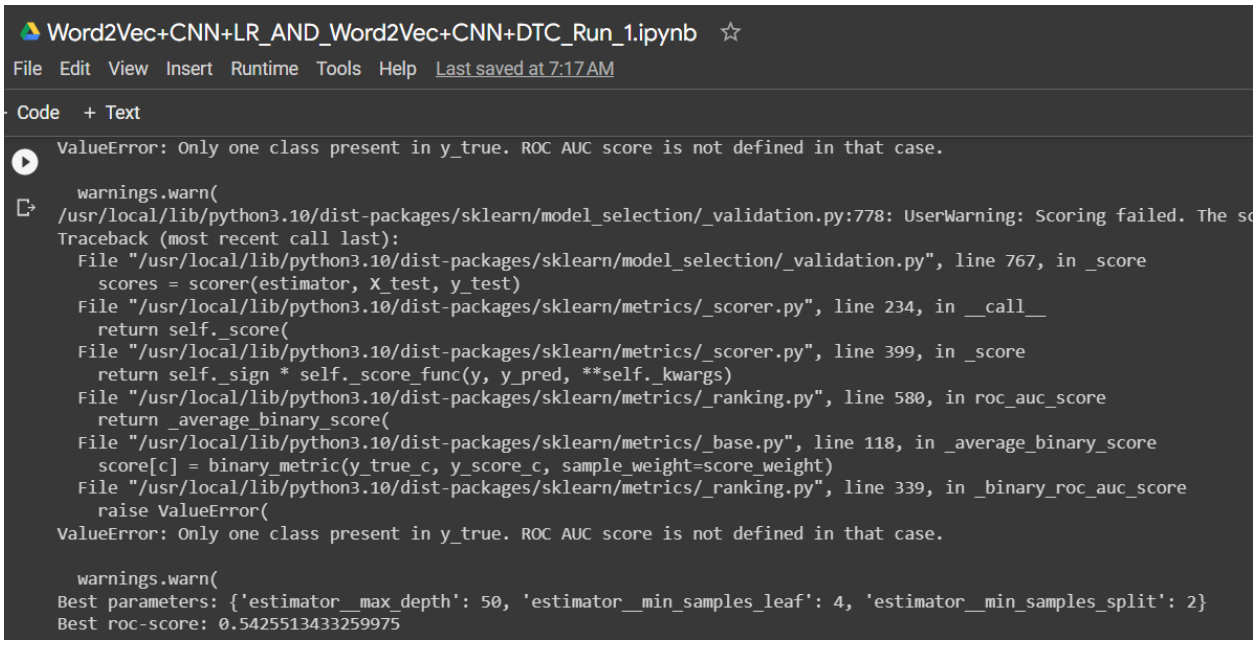


3d. Test results for Word 2 vec +CNN +LR run 2



4. Word 2 vec +CNN +DTC

4a. Parameter Tuning for Word 2 Vec + CNN +DTC run1



4a.Test Results for Word 2 Vec + CNN +DTC run1

Word2Vec+CNN+LR_AND_Word2Vec+CNN+DTC_Run_1.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 7:17 AM

Code + Text

```
[ ]    else:
         auc_roc=0
         average_precision=0
       acc=accuracy_score(train_Y_good[i],y_pred.reshape(-1,1))
       PR_AUC.append(average_precision)
       ACC.append(acc)
       ROC_AUC.append(auc_roc)

   # Print evaluation metrics
   print('ACC score is',sum(ACC)/len(ACC))
   print('ROC_AUC score is',sum(ROC_AUC)/len(ROC_AUC))
   print('PR_AUC score is',sum(PR_AUC)/len(PR_AUC))

   # Measure execution time
   %time
```

```
ACC score is 0.9191237113402058
ROC_AUC score is 0.7482500903647535
PR_AUC score is 0.21909921863710352
CPU times: user 1e+03 ns, sys: 2 µs, total: 3 µs
Wall time: 6.68 µs
```

4c. Parameter Tuning for Word 2 Vec + CNN +DTC run 2

Word2Vec+CNN+LR_AND_Word2Vec+CNN+DTC_Run_2.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 7:26 AM

Code + Text

```
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_validation.py:778: UserWarning: Scoring failed. The score on this tr
Traceback (most recent call last):
  File "/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_validation.py", line 767, in _score
    scores = scorer(estimator, X_test, y_test)
  File "/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_scorer.py", line 234, in __call__
    return self._score(
  File "/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_scorer.py", line 399, in _score
    return self._sign * self._score_func(y, y_pred, **self._kwargs)
  File "/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_ranking.py", line 580, in roc_auc_score
    return _average_binary_score(
  File "/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_base.py", line 118, in _average_binary_score
    score[c] = binary_metric(y_true_c, y_score_c, sample_weight=score_weight)
  File "/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_ranking.py", line 339, in _binary_roc_auc_score
    raise ValueError(
ValueError: Only one class present in y_true. ROC AUC score is not defined in that case.

  warnings.warn(
Best parameters: {'estimator__max_depth': 10, 'estimator__min_samples_leaf': 4, 'estimator__min_samples_split': 2}
Best roc-score: 0.5425730019062781
```

4d. Test Results for Word 2 Vec + CNN +DTC run 2

Word2Vec+CNN+LR_AND_Word2Vec+CNN+DTC_Run_2.ipynb ☆

File Edit View Insert Runtime Tools Help Last saved at 7:26 AM

Code + Text

```
    acc=accuracy_score(train_Y_good[1],y_pred.reshape(-1,1))
    PR_AUC.append(average_precision)
    ACC.append(acc)
    ROC_AUC.append(auc_roc)

# Print evaluation metrics
print('ACC score is',sum(ACC)/len(ACC))
print('ROC_AUC score is',sum(ROC_AUC)/len(ROC_AUC))
print('PR_AUC score is',sum(PR_AUC)/len(PR_AUC))

# Measure execution time
%time
```

```
ACC score is 0.9191237113402058
ROC_AUC score is 0.7482516904533314
PR_AUC score is 0.2192505143688108
CPU times: user 1e+03 ns, sys: 2 µs, total: 3 µs
Wall time: 5.25 µs
```