REPORT PROJECT 2

Name: Leela Veera NagendraReddy Padala
Zid: Z5249335

Task-2

1) F1 score 0.7483312619309965

```
69
70 # task 1.3
71 pred_test = test_prediction(test_data, base_features_pipeline_model, gen_base_pred_pipeline_model, gen_meta_feature_pipeline
72
73 # Evaluation
74 evaluator = MulticlassClassificationEvaluator(predictionCol="prediction",metricName='f1')
75 print(evaluator.evaluate(pred_test, {evaluator.predictionCol:'final_prediction'}))
76 spark.stop()
```

```
0.7483312619309965
```

2) For Improving the ML algorithms accuracy or F1 Score , we need to make sure that Features selected for Training our Model is error free and we can use Tuning Features for finding the best F1 score .

Steps I used to Increase the F1 score:

1)I had deleted all the stop words by Using StopWordsRemover module .

We need to use this StopWordsRemover after word_tokenizer as shown below:

```
def base_features_gen_pipeline(input_descript_col="descript", input_category_col="category", outpu
output_label_col="label"):
    word_tokenizer = Tokenizer(inputCol=input_descript_col, outputCol="words")
    stopwords=StopWordsRemover(inputCol="words",outputCol="Nostopwords")
    count_vectors = CountVectorizer(inputCol="Nostopwords", outputCol=output_feature_col)
    label_maker = StringIndexer(inputCol = input_category_col, outputCol = output_label_col)
    selector = Selector(outputCols = ['id','features', 'label'])
    return Pipeline(stages=[word_tokenizer,stopwords, count_vectors, label_maker, selector])
```

Now the Pipeline will send output without any Stop Words. But accuracy is not improved alone by this

2)There comes Tuning of Parameters while training the model . For Finding best Tuning parameters , we need to see what classifiers we are using.

In this Project we are using NBClassifier, SVMClassifier and Logistic Regression as our Algorithms.

For NB classifier , smoothing parameter can be varied and we can use Best smoothing. For finding this create an array with all the possible values and send the smoothing value one by one and use the best smoothing value

nb_0 = NaiveBayes(featuresCol='features', labelCol='label_0', predictionCol='nb_pred_0', probabilityCol='nb_prob_0', rawPredictionCol='nb_raw_0',smoothing=0.2)

nb_1 = NaiveBayes(featuresCol='features', labelCol='label_1', predictionCol='nb_pred_1', probabilityCol='nb_prob_1', rawPredictionCol='nb_raw_1',smoothing=0.2)

nb_2 = NaiveBayes(featuresCol='features', labelCol='label_2', predictionCol='nb_pred_2', probabilityCol='nb_prob_2', rawPredictionCol='nb_raw_2',smoothing=0.2)

svm_0 = LinearSVC(featuresCol='features', labelCol='label_0', predictionCol='svm_pred_0', rawPredictionCol='svm_raw_0',maxIter=90)

svm_1 = LinearSVC(featuresCol='features', labelCol='label_1', predictionCol='svm_pred_1', rawPredictionCol='svm_raw_1',maxIter=90)

svm_2 = LinearSVC(featuresCol='features', labelCol='label_2', predictionCol='svm_pred_2', rawPredictionCol='svm_raw_2',maxIter=90)

lr_model = LogisticRegression(featuresCol='meta_features', labelCol='label', predictionCol='final_prediction', maxIter=20, regParam=1., elasticNetParam=0)

meta_classifier = lr_model.fit(meta_features)

By Changing the Smoothing =0.2 and maxIter=90 in LinearSVC .

I ACHIEVED 0.750159 accuracy.