1. **Data Processing Steps:**

    1. List of data issues found in the raw data.:There were some issues with the raw data provided mainly:

- Junk Values
- Column selection
- Null Values
- Invalid values in gender column
- Selecting data based on gender confidence
- Merging fields description and text

    2. How did you tackle the issues mentioned above?

- **Junk Values** — This was taken care by using option("mode", “DROPMALFORMED") which basically ignores the whole corrupted records while parsing

```
Dataset<Row> dataset = sparkSession.read().format("csv").option("header","true").option("mode", "DROPMALFORMED").load(path);
```

- **Column Selection:** We only took Gender, gender:confidence, text and description column as other columns won’t be utilised in our calculations

```
Dataset<Row> testdata = testDat.select(testDat.col("description"), testDat.col("gender"),

testDat.col("gender:confidence"),testDat.col("text"));
```

- **Null Values:** — This was taken care by .na().drop() which will ignore the rows having Null data.This step was don’t after the selection of two columns Gender and Description as there are columns in the sheet where all the values are null hence whole dataset will get ignored.

```
Dataset<Row> datasetCleantestda =
datasetCleantest.na().drop();
```

- **merging text and description:** To get more data for the model we merged text and description.

```
Dataset<Row> datasetCleantes =
datasetCleantestda.withColumn("descriptio",
concat(datasetCleantestda.col("description"),
                        lit(" "),datasetCleantestda.col("text")));
Dataset<Row> datasetCleantestdat =
datasetCleantes.select(datasetCleantes.col("gender"),

datasetCleantes.col("gender:confidence"),datasetCleantes.col("des
criptio"));
```

- **Elimination of invalid values in Gender column:** This has been taken care by .where("gender = 'male' OR gender ='female' OR gender = ‘brand"') which will ignore rows which has any other values as those records are of no use to us because of the invalid values

```
Dataset<Row> datasetCleantestda =
datasetCleantest.na().drop();
```

- **Data based on gender confidence:**

```
Dataset<Row> datasetCleantestdata =
datasetCleantestdat.where(col("gender:confidence").$greater(.
99999));
```

2. **Model Building :**
1. You need to clearly list all the different models you have used.

- Decision tree:

```
DecisionTreeClassifier dt = new DecisionTreeClassifier();
```

- Random forests:

```
RandomForestClassifier rf = new RandomForestClassifier();
```

2. For each model, clearly explain what data (columns) is being used to train the model.

For both the models we have used:

Indexer:

```
StringIndexerModel labelindexer = new StringIndexer()
                        .setInputCol("gender")
                        .setOutputCol(“label”).fit(datasetCleantraindata)
;
```

Tokenizer: This outputs to column words

```
Tokenizer tokenizer = new Tokenizer()
                        .setInputCol("descriptio")
                        .setOutputCol("words");
```

Hashing Term Frequency Matrix: This outputs to column numFeatures

```
    HashingTF hashingTF = new HashingTF()
                        .setNumFeatures(1000)
                        .setInputCol(tokenizer.getOutputCol())
                        .setOutputCol(“numFeatures");
```

Inverse Document Frequency: This outputs to column features

```
IDF idf = new IDF()
                        .setInputCol(hashingTF.getOutputCol())
                        .setOutputCol("features");
```

Decision Tree Implementation:

```
Create and Run Decision Tree Pipeline
        Pipeline pipelineDT = new Pipeline()
                .setStages(new PipelineStage[] {labelindexer,
tokenizer,hashingTF, idf, dt,labelConverter});
        // Fit the pipeline to training documents.
        PipelineModel modelDT =
pipelineDT.fit(datasetCleantraindata);
        // Make predictions on test documents.
        Dataset<Row> predictionsDT =
modelDT.transform(datasetCleantestdata);
        System.out.println("Predictions from Decision Tree Model
are:");
        predictionsDT.show(10);
```

Random Forest Implementation:

```
// Create and Run Random Forest Pipeline
        Pipeline pipelineRF = new Pipeline()
                .setStages(new PipelineStage[] {labelindexer,
tokenizer, hashingTF, idf, rf,labelConverter});
        // Fit the pipeline to training documents.
        PipelineModel modelRF =
pipelineRF.fit(datasetCleantraindata);
        // Make predictions on test documents.
        Dataset<Row> predictionsRF =
modelRF.transform(datasetCleantestdata);
        System.out.println("Predictions from Random Forest
Model are:");
        predictionsRF.show(10);
```

3. **Evaluation Metrics:**
    1. For each model, you need to report the following performance metrics:

- Evaluation Scores:

Accuracy:

```
Accuracy for Random Forest = 0.4729555698647778
Accuracy for Decision Tree = 0.4858338699291693
```

F-Score:

```
fscore for Random forest= 0.37521351999620867

fscore for Decision Tree= 0.4099726248702883
```

Confusion matrix:

```
Confusion Matrix for Random Forest:
+-----+--------------+-----+
|label|predictedLabel|count|
+-----+--------------+-----+
|  0.0|         brand|    9|
|  2.0|        female|  510|
|  0.0|          male|   21|
|  0.0|        female| 1223|
|  2.0|         brand|  178|
|  2.0|          male|   63|
|  1.0|        female| 1020|
|  1.0|          male|   68|
|  1.0|         brand|   14|
+-----+--------------+-----+



Confusion Matrix for Decision Tree:
Confusion Matrix for Decision Tree:
+-----+--------------+-----+
|label|predictedLabel|count|
+-----+--------------+-----+
|  0.0|         brand|  161|
|  2.0|        female|  322|
|  0.0|          male|   66|
|  0.0|        female| 1026|
|  2.0|         brand|  418|
```

```
|  2.0|          male|   11|
|  1.0|        female|  837|
|  1.0|          male|   65|
|  1.0|         brand|  200|
+-----+--------------+-----+
```

2. For each model, check if the model is facing any issue of overfitting or underfitting:

- Since the output of train and test data is almost similar we can term it all **underfitting** scenario

Train Data output:

```
Accuracy for Random Forest = 0.48429951690821255

Accuracy for Decision Tree = 0.4502415458937198
```

Test Data Output:

```
Accuracy for Random Forest = 0.48412442148315576

Accuracy for Decision Tree = 0.4763749865461199
```

4. **Inferences & Suggestions:**

1. Compare the results from the models and mention drawbacks and advantages for each of them:

I tried running the same model by using text and colour as columns but the accuracy seems to be almost same as the accuracy i got from description column, hence we might need to combine columns in case we want to improve accuracy.

2. Suggest some improvisation techniques to those models.

Combine multiple columns like sidebar_color, description, text to increase the accuracy of the model.

3. Choose from the two models present and justify why did you choose that particular model for the given problem statement

I chose Decision tree algorithm as model because the accuracy is more as compared to Decision tree.

```
Accuracy for Random Forest = 0.47
Accuracy for Decision Tree = 0.49
```

Complete Console output:

```
Using Spark's default log4j profile: org/apache/spark/log4j-
defaults.properties
Predictions from Random Forest Model are:
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+--------------+
|gender|gender:confidence|          descriptio|label|
words|         numFeatures|            features|
rawPrediction|         probability|prediction|predictedLabel|
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+--------------+
|female|                1|you don't know me...|  0.0|[you, don't,
know...|(1000,[82,101,106...|(1000,[82,101,106...|
[7.87959442011705...|[0.39397972100585...|       0.0|
female|
| brand|                1|A global marketpl...|  2.0|[a, global,
marke...|(1000,[36,82,107,...|(1000,[36,82,107,...|
[6.86186482408806...|[0.34309324120440...|       1.0|
male|
|female|                1|JMKM?_ҕ?? @gianna...|  0.0|[jmkm?_ҕ??,
@gian...|(1000,[36,53,96,1...|(1000,[36,53,96,1...|
[7.51797789285827...|[0.37589889464291...|       0.0|
female|
|female|                1|He bled and died ...|  0.0|[he, bled,
and, d...|(1000,[142,148,21...|(1000,[142,148,21...|
[9.14319965099619...|[0.45715998254980...|       0.0|
female|
|  male|                1|Militante y obrer...|  1.0|[militante,
y, ob...|(1000,[7,8,29,32,...|(1000,[7,8,29,32,...|
[8.37014864305956...|[0.41850743215297...|       0.0|
female|
```

```
|  male|                1|[ Krothedj@gmail....|  1.0|[[,
krothedj@gmai...|(1000,[106,206,27...|(1000,[106,206,27...|
[7.33559594629007...|[0.36677979731450...|       0.0|
female|
|  male|                1|Just Living Life ...|  1.0|[just,
living, li...|(1000,[56,119,168...|(1000,[56,119,168...|
[7.8664885534634,...|[0.39332442767317...|       0.0|
female|
|  male|                1|Just a curious gu...|  1.0|[just, a,
curious...|(1000,[94,170,307...|(1000,[94,170,307...|
[7.91817824079432...|[0.39590891203971...|       0.0|
female|
|female|                1|ask no questions ...|  0.0|[ask, no,
questio...|(1000,[44,114,138...|(1000,[44,114,138...|
[7.41253428201862...|[0.37062671410093...|       0.0|
female|
|  male|                1|For any photograp...|  1.0|[for, any,
photog...|(1000,[4,36,66,77...|(1000,[4,36,66,77...|
[7.44770864595261...|[0.37238543229763...|       0.0|
female|
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+-------------+
only showing top 10 rows

Predictions from Decision Tree Model are:
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+-------------+
|gender|gender:confidence|          descriptio|label|
words|         numFeatures|            features|
rawPrediction|         probability|prediction|predictedLabel|
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+-------------+
|female|                1|you don't know me...|  0.0|[you, don't,
know...|(1000,[82,101,106...|(1000,[82,101,106...|
[929.0,907.0,641.0]|[0.37505046427129...|       0.0|
female|
| brand|                1|A global marketpl...|  2.0|[a, global,
marke...|(1000,[36,82,107,...|(1000,[36,82,107,...|
[228.0,304.0,454.0]|[0.23123732251521...|       2.0|
brand|
|female|                1|JMKM?_ҕ?? @gianna...|  0.0|[jmkm?_ҕ??,
@gian...|(1000,[36,53,96,1...|(1000,[36,53,96,1...|
[228.0,304.0,454.0]|[0.23123732251521...|       2.0|
brand|
```

```
|female|                1|He bled and died ...|  0.0|[he, bled,
and, d...|(1000,[142,148,21...|(1000,[142,148,21...|
[265.0,144.0,40.0]|[0.59020044543429...|       0.0|        female|
|  male|                1|Militante y obrer...|  1.0|[militante,
y, ob...|(1000,[7,8,29,32,...|(1000,[7,8,29,32,...|
[965.0,609.0,95.0]|[0.57819053325344...|       0.0|        female|
|  male|                1|[ Krothedj@gmail....|  1.0|[[,
krothedj@gmai...|(1000,[106,206,27...|(1000,[106,206,27...|
[929.0,907.0,641.0]|[0.37505046427129...|       0.0|
female|
|  male|                1|Just Living Life ...|  1.0|[just,
living, li...|(1000,[56,119,168...|(1000,[56,119,168...|
[929.0,907.0,641.0]|[0.37505046427129...|       0.0|
female|
|  male|                1|Just a curious gu...|  1.0|[just, a,
curious...|(1000,[94,170,307...|(1000,[94,170,307...|
[929.0,907.0,641.0]|[0.37505046427129...|       0.0|
female|
|female|                1|ask no questions ...|  0.0|[ask, no,
questio...|(1000,[44,114,138...|(1000,[44,114,138...|
[929.0,907.0,641.0]|[0.37505046427129...|       0.0|
female|
|  male|                1|For any photograp...|  1.0|[for, any,
photog...|(1000,[4,36,66,77...|(1000,[4,36,66,77...|
[228.0,304.0,454.0]|[0.23123732251521...|       2.0|
brand|
+------+----------------+--------------------+-----
+--------------------+--------------------+--------------------
+--------------------+--------------------+----------
+--------------+
only showing top 10 rows

Accuracy for Random Forest = 0.4729555698647778
Accuracy for Decision Tree = 0.4858338699291693
Confusion Matrix for Random Forest:
+-----+--------------+-----+
|label|predictedLabel|count|
+-----+--------------+-----+
|  0.0|         brand|    9|
|  2.0|        female|  510|
|  0.0|          male|   21|
|  0.0|        female| 1223|
|  2.0|         brand|  178|
|  2.0|          male|   63|
|  1.0|        female| 1020|
|  1.0|          male|   68|
|  1.0|         brand|   14|
+-----+--------------+-----+

fscore for Random forest= 0.37521351999620867
Confusion Matrix for Decision Tree:
+-----+--------------+-----+
```

```
|label|predictedLabel|count|
+-----+--------------+-----+
|  0.0|         brand|  161|
|  2.0|        female|  322|
|  0.0|          male|   66|
|  0.0|        female| 1026|
|  2.0|         brand|  418|
|  2.0|          male|   11|
|  1.0|        female|  837|
|  1.0|          male|   65|
|  1.0|         brand|  200|
+-----+--------------+-----+

fscore for Decision Tree= 0.4099726248702883
```