Detect data bias with Amazon SageMaker Clarify

Introduction

Bias can be present in your data before any model training occurs. Inspecting the dataset for bias can help detect collection gaps, inform your feature engineering, and understand societal biases the dataset may reflect. In this lab you will analyze bias on the dataset, generate and analyze bias report, and prepare the dataset for the model training.

Table of Contents

First, let's install and import required modules.

In [1]:

```
# please ignore warning messages during the installation
!pip install --disable-pip-version-check -q sagemaker==2.35.0
```

```
/opt/conda/lib/python3.7/site-packages/secretstorage/dhcrypto.py:16: CryptographyDeprecat
ionWarning: int_from_bytes is deprecated, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
/opt/conda/lib/python3.7/site-packages/secretstorage/util.py:25: CryptographyDeprecationW
arning: int_from_bytes is deprecated, use int.from_bytes instead
  from cryptography.utils import int_from_bytes
WARNING: Running pip as root will break packages and permissions. You should install pack
ages reliably by using venv: https://pip.pypa.io/warnings/venv
```

In [2]:

```
import boto3
import sagemaker
import pandas as pd
import numpy as np

sess   = sagemaker.Session()
```

```
bucket = sess.default_bucket()
role = sagemaker.get_execution_role()
region = boto3.Session().region_name
```

In [3]:

```
import matplotlib.pyplot as plt
%matplotlib inline
%config InlineBackend.figure_format='retina'
```

1. Analyze the dataset

1.1. Create a pandas data frame from the CSV file

Create a pandas dataframe from each of the product categories and concatenate them into one.

In [4]:

```
!aws s3 cp 's3://dlai-practical-data-science/data/transformed/womens_clothing_ecommerce_reviews_transformed.csv' ./
```

```
download: s3://dlai-practical-data-science/data/transformed/womens_clothing_ecommerce_reviews_transformed.csv to ./womens_clothing_ecommerce_reviews_transformed.csv
```

In [5]:

```
path = './womens_clothing_ecommerce_reviews_transformed.csv'

df = pd.read_csv(path)
df.head()
```

Out[5]:

	sentiment	review_body	product_category
0	1	If this product was in petite i would get the...	Blouses
1	1	Love this dress! it's sooo pretty. i happene...	Dresses
2	0	I had such high hopes for this dress and reall...	Dresses
3	1	I love love love this jumpsuit. it's fun fl...	Pants
4	1	This shirt is very flattering to all due to th...	Blouses

As you saw in the previous lab, there are way more positive reviews than negative or neutral. Such a dataset is called unbalanced.

In this case, using a relatively small data subset you could visualize the occurring unbalances. At scale, you would need to perform bias analysis. Let's use this dataset as an example.

In [6]:

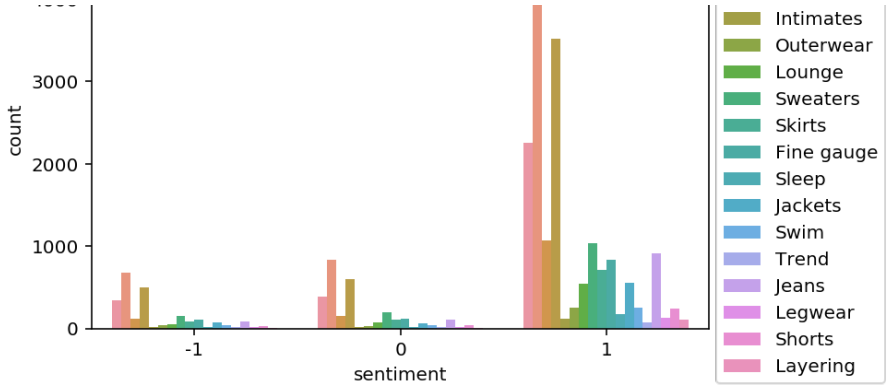
```
import seaborn as sns

sns.countplot(data=df, x='sentiment', hue='product_category')

plt.legend(loc='upper right',bbox_to_anchor=(1.3, 1.1))
```

Out[6]:

```
<matplotlib.legend.Legend at 0x7f7d1f36f050>
```



1.2. Upload the dataset to S3 bucket

Upload the dataset to a private S3 bucket in a folder called `bias/unbalanced`.

In [7]:

```
data_s3_uri_unbalanced = sess.upload_data(bucket=bucket,
                                          key_prefix='bias/unbalanced',
                                          path='./womens_clothing_ecommerce_reviews_transformed.csv
')
data_s3_uri_unbalanced
```

Out[7]:

```
's3://sagemaker-us-east-1-390574811984/bias/unbalanced/womens_clothing_ecommerce_reviews_
transformed.csv'
```

You can review the uploaded CSV file in the S3 bucket.

Instructions:

- **open the link**
- **click on the S3 bucket name** `sagemaker-us-east-1-ACCOUNT`
- **go to the folder** `bias/unbalanced`
- **check the existence of the file** `womens_clothing_ecommerce_reviews_transformed.csv`

In [8]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="top" href="https://s3.console.aws.amazon.com/s3/home?r
egion={}#"">Amazon S3 bucket</a></b>'.format(region)))
```

Review Amazon S3 bucket

2. Analyze class imbalance on the dataset with Amazon SageMaker Clarify

Let's analyze bias in `sentiment` with respect to the `product_category` facet on the dataset.

2.1. Configure a `DataConfig`

Information about the input data needs to be provided to the processor. This can be done with the `DataConfig` of the Clarify container. It stores information about the dataset to be analyzed, for example the dataset file, its format, headers and labels.

Exercise 1

Configure a `DataConfig` for Clarify.

Instructions: Use `DataConfig` to configure the target column (`'sentiment'` label), data input (`data_s3_uri_unbalanced`) and output paths (`bias_report_unbalanced_output_path`) with their formats (header names and the dataset type):

```
    data_config_unbalanced = clarify.DataConfig(
        s3_data_input_path=..., # S3 object path containing the unbalanced dataset
        s3_output_path=..., # path to store the output
        label='...', # target column
        headers=df_unbalanced.columns.to_list(),
        dataset_type='text/csv'
    )
```

In [14]:

```
from sagemaker import clarify

bias_report_unbalanced_output_path = 's3://{}/bias/generated_bias_report/unbalanced'.form
at(bucket)

data_config_unbalanced = clarify.DataConfig(
    ### BEGIN SOLUTION - DO NOT delete this comment for grading purposes
    s3_data_input_path=data_s3_uri_unbalanced, # Replace None
    s3_output_path=bias_report_unbalanced_output_path, # Replace None
    label='sentiment', # Replace None
    ### END SOLUTION - DO NOT delete this comment for grading purposes
    headers=df.columns.to_list(),
    dataset_type='text/csv'
)
```

2.2. Configure `BiasConfig`

Bias is measured by calculating a metric and comparing it across groups. To compute it, you will specify the required information in the `BiasConfig` API. SageMaker Clarify needs the sensitive columns (`facet_name`) and the desirable outcomes (`label_values_or_threshold`). Here `product_category` is the sensitive facet and the desired outcome is with the `sentiment==1` .

SageMaker Clarify can handle both categorical and continuous data for `label_values_or_threshold` . In this case you are using categorical data.

In [15]:

```
bias_config_unbalanced = clarify.BiasConfig(
    label_values_or_threshold=[1], # desired sentiment
    facet_name='product_category' # sensitive column (facet)
)
```

2.3. Configure Amazon SageMaker Clarify as a processing job

Now you need to construct an object called `SageMakerClarifyProcessor` . This allows you to scale the process of data bias detection using two parameters, `instance_count` and `instance_type` . `Instance_count` represents how many nodes you want in the distributor cluster during the data detection. `Instance_type` specifies the processing capability (compute capacity, memory capacity) available for each one of those nodes.

In [16]:

```
clarify_processor_unbalanced = clarify.SageMakerClarifyProcessor(role=role,
                                                                 instance_count=1,
                                                                 instance_type='ml.m5.large',
                                                                 sagemaker_session=sess)
```

2.4. Run the Amazon SageMaker Clarify processing job

Exercise 2

Run the configured processing job to compute the requested bias `methods` of the input data

Instructions: Apply the `run_pre_training_bias` method to the configured Clarify processor, passing the configured input/output data (`data_config_unbalanced`), configuration of sensitive groups (`bias_config_unbalanced`) with the other job setup parameters:

```
clarify_processor_unbalanced.run_pre_training_bias(
    data_config=..., # configured input/output data
    data_bias_config=..., # configured sensitive groups
    methods=["CI", "DPL", "KL", "JS", "LP", "TVD", "KS"], # selector of a subset of potential metrics
    wait=False, # whether the call should wait until the job completes (default: True)
    logs=False # whether to show the logs produced by the job. Only meaningful when wait is True (default: True)
)
```

In [17]:

```
clarify_processor_unbalanced.run_pre_training_bias(
    ### BEGIN SOLUTION - DO NOT delete this comment for grading purposes
    data_config=data_config_unbalanced, # Replace None
    data_bias_config=bias_config_unbalanced, # Replace None
    ### END SOLUTION - DO NOT delete this comment for grading purposes
    methods=["CI", "DPL", "KL", "JS", "LP", "TVD", "KS"],
    wait=False,
    logs=False
)
```

```
Job Name:  Clarify-Pretraining-Bias-2021-06-25-15-52-31-970
Inputs:  [{'InputName': 'dataset', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagemaker-us-east-1-390574811984/bias/unbalanced/womens_clothing_ecommerce_reviews_transformed.csv', 'LocalPath': '/opt/ml/processing/input/data', 'S3DataType': 'S3Prefix', 'S3InputMode': 'File', 'S3DataDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}, {'InputName': 'analysis_config', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/analysis_config.json', 'LocalPath': '/opt/ml/processing/input/config', 'S3DataType': 'S3Prefix', 'S3InputMode': 'File', 'S3DataDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}]
Outputs:  [{'OutputName': 'analysis_result', 'AppManaged': False, 'S3Output': {'S3Uri': 's3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced', 'LocalPath': '/opt/ml/processing/output', 'S3UploadMode': 'EndOfJob'}}]
```

In [18]:

```
run_unbalanced_bias_processing_job_name = clarify_processor_unbalanced.latest_job.job_name
print(run_unbalanced_bias_processing_job_name)
```

```
Clarify-Pretraining-Bias-2021-06-25-15-52-31-970
```

2.5. Run and review the Amazon SageMaker Clarify processing job on the unbalanced dataset

Review the created Amazon SageMaker Clarify processing job and the Cloud Watch logs.

Instructions:

- **open the link**
- **note that you are in the section Amazon SageMaker -> Processing jobs**
- **check the processing job name**
- **note which other properties of the processing job you can see in the console**

In [19]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="https://console.aws.amazon.com/sagemaker/
home?region={}#/processing-jobs/{}">processing job</a></b>'.format(region, run_unbalanced
_bias_processing_job_name)))
```

Review processing job

Instructions:

- **open the link**
- **open the log stream with the name, which starts from the processing job name**
- **have a quick look at the log messages**

In [20]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="https://console.aws.amazon.com/cloudwatch
/home?region={}#logStream:group=/aws/sagemaker/ProcessingJobs;prefix={};streamFilter=type
LogStreamPrefix">CloudWatch logs</a> after about 5 minutes</b>'.format(region, run_unbala
nced_bias_processing_job_name)))
```

Review CloudWatch logs after about 5 minutes

In [21]:

```
running_processor = sagemaker.processing.ProcessingJob.from_processing_name(processing_jo
b_name=run_unbalanced_bias_processing_job_name,
                                                                             sagemaker_s
ession=sess)
```

This cell will take approximately 5-10 minutes to run.

In [22]:

```
%%time

running_processor.wait(logs=False)
```

```
......!CPU times: user 44.3 ms, sys: 4.44 ms, total: 48.7 ms
Wall time: 30.3 s
```

2.6. Analyze unbalanced bias report

In this run, you analyzed bias for `sentiment` relative to the `product_category` for the unbalanced data. Let's have a look at the bias report.

List the files in the output path `bias_report_unbalanced_output_path`:

In [23]:

```
!aws s3 ls $bias_report_unbalanced_output_path/
```

```
2021-06-25 15:57:49      31732 analysis.json
2021-06-25 15:52:33        346 analysis_config.json
2021-06-25 15:57:49     387134 report.html
2021-06-25 15:57:49     121999 report.ipynb
2021-06-25 15:57:49     139371 report.pdf
```

Download generated bias report from S3 bucket:

In [24]:

```
!aws s3 cp --recursive $bias_report_unbalanced_output_path ./generated_bias_report/unbal
anced/
```

```
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/ana
lysis_config.json to generated_bias_report/unbalanced/analysis_config.json
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/ana
lysis.json to generated_bias_report/unbalanced/analysis.json
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/rep
ort.ipynb to generated_bias_report/unbalanced/report.ipynb
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/rep
ort.html to generated_bias_report/unbalanced/report.html
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/unbalanced/rep
ort.pdf to generated_bias_report/unbalanced/report.pdf
```

Review the downloaded bias report (in HTML format):

In [25]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="./generated_bias_report/unbalanced/report
.html">unbalanced bias report</a></b>'))
```

Review unbalanced bias report

The bias report shows a number of metrics, but here you can focus on just two of them:

- **Class Imbalance (CI). Measures the imbalance in the number of members between different facet values. Answers the question, does a `product_category` have disproportionately more reviews than others? Values of CI will become equal for even distribution between facets. Here, different CI values show the existence of imbalance.**
- **Difference in Positive Proportions in Labels (DPL). Measures the imbalance of positive outcomes between different facet values. Answers the question, does a `product_category` have disproportionately higher ratings than others? With the range over the interval from -1 to 1, if there is no bias, you want to see this value as close as possible to zero. Here, non-zero values indicate the imbalances.**

3. Balance the dataset by `product_category` and `sentiment`

Let's balance the dataset by `product_category` and `sentiment`. Then you can configure and run SageMaker Clarify processing job to analyze the bias of it. Which metrics values do you expect to see in the bias report?

In [26]:

```
df_grouped_by = df.groupby(['product_category', 'sentiment'])
df_balanced = df_grouped_by.apply(lambda x: x.sample(df_grouped_by.size().min()).reset_in
dex(drop=True))
```

In [27]:

```
df_balanced
```

Out[27]:

			sentiment	review_body	product_category
product_category	**sentiment**				
Blouses	**-1**	**0**	**-1**	**This is a gorgeous top. very well made with a ...**	**Blouses**
		1	**-1**	**I really wanted to like this top just got it ...**	**Blouses**
		2	**-1**	**The concept of this top is cool and the outer ...**	**Blouses**
		3	**-1**	**I'm normally a small in retailer tops. i order...**	**Blouses**
		4	**-1**	**Didn't work on this curvy gal. sizing up woul...**	**Blouses**
...	**...**	**...**	**...**	**...**	**...**
Trend	**1**	**4**	**1**	**I tried on a size 8 in the store just for fun....**	**Trend**
		5	**1**	**This is an adorably gorgeous swing dress. the ...**	**Trend**
		6	**1**	**This dress is much larger in person than it lo...**	**Trend**
		7	**1**	**Swing dresses are the holy grail of comfort. w...**	**Trend**
		8	**1**	**This dress is very pretty and would be nice fo...**	**Trend**

486 rows × 3 columns

Visualize the distribution of review sentiment in the balanced dataset.

In [28]:

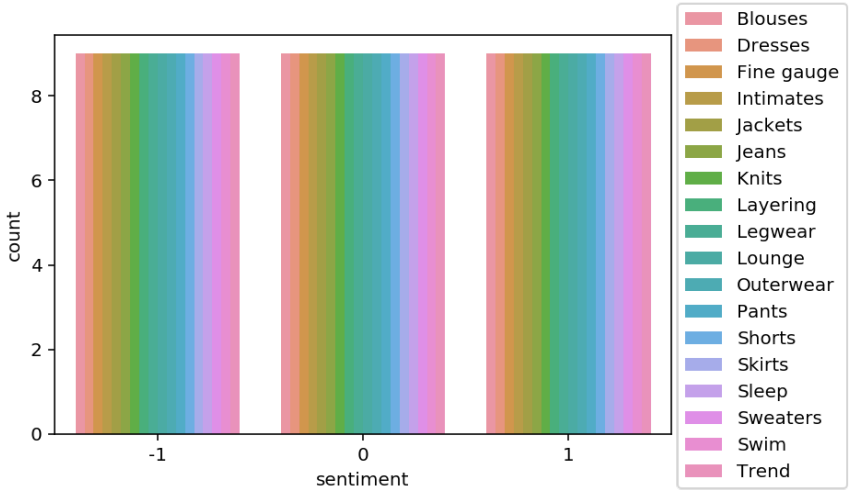
```
import seaborn as sns

sns.countplot(data=df_balanced, x='sentiment', hue='product_category')

plt.legend(loc='upper right',bbox_to_anchor=(1.3, 1.1))
```

Out[28]:

```
<matplotlib.legend.Legend at 0x7f7d1677d390>
```



4. Analyze bias on balanced dataset with Amazon SageMaker Clarify

Let's analyze bias in `sentiment` with respect to the `product_category` facet on your balanced dataset.

Save and upload balanced data to S3 bucket.

In [29]:

```
path_balanced = './womens_clothing_ecommerce_reviews_balanced.csv'
```

```
df_balanced.to_csv(path_balanced, index=False, header=True)

data_s3_uri_balanced = sess.upload_data(bucket=bucket, key_prefix='bias/balanced', path=
path_balanced)
data_s3_uri_balanced
```

Out[29]:

```
's3://sagemaker-us-east-1-390574811984/bias/balanced/womens_clothing_ecommerce_reviews_ba
lanced.csv'
```

You can review the uploaded CSV file in the S3 bucket and prefix `bias/balanced`.

In [30]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="top" href="https://s3.console.aws.amazon.com/s3/home?r
egion={}#">Amazon S3 bucket</a></b>'.format(region)))
```

Review Amazon S3 bucket

4.1. Configure a `DataConfig`

Exercise 3

Configure a `DataConfig` for Clarify to analyze bias on the balanced dataset.

Instructions: Pass the S3 object path containing the balanced dataset, the path to store the output (`bias_report_balanced_output_path`) and the target column. You can use exercise 1 as an example.

In [31]:

```
from sagemaker import clarify

bias_report_balanced_output_path = 's3://{}/bias/generated_bias_report/balanced'.format(b
ucket)

data_config_balanced = clarify.DataConfig(
    ### BEGIN SOLUTION - DO NOT delete this comment for grading purposes
    s3_data_input_path=data_s3_uri_balanced, # Replace None
    s3_output_path=bias_report_balanced_output_path, # Replace None
    label='sentiment', # Replace None
    ### END SOLUTION - DO NOT delete this comment for grading purposes
    headers=df_balanced.columns.to_list(),
    dataset_type='text/csv'
)
```

4.2. Configure `BiasConfig`

`BiasConfig` for the balanced dataset will have the same settings as before.

In [32]:

```
bias_config_balanced = clarify.BiasConfig(
    label_values_or_threshold=[1], # desired sentiment
    facet_name='product_category' # sensitive column (facet)
)
```

4.3. Configure SageMaker Clarify as a processing job

`SageMakerClarifyProcessor` object will also have the same parameters.

In [33]:

```
clarify_processor_balanced = clarify.SageMakerClarifyProcessor(role=role,
                                                               instance_count=1,
                                                               instance_type='ml.m5.large',
                                                               sagemaker_session=sess)
```

4.4. Run the Amazon SageMaker Clarify processing job

Exercise 4

Run the configured processing job for the balanced dataset.

Instructions: Apply the `run_pre_training_bias` method to the configured Clarify processor, passing the input/output data, configuration of sensitive groups with the other job setup parameters. You can use exercise 2 as an example.

In [34]:

```
clarify_processor_balanced.run_pre_training_bias(
    ### BEGIN SOLUTION - DO NOT delete this comment for grading purposes
    data_config=data_config_balanced, # Replace None
    data_bias_config=bias_config_balanced, # Replace None
    ### END SOLUTION - DO NOT delete this comment for grading purposes
    methods=["CI", "DPL", "KL", "JS", "LP", "TVD", "KS"],
    wait=False,
    logs=False
)
```

```
Job Name:  Clarify-Pretraining-Bias-2021-06-25-16-05-21-917
Inputs:  [{'InputName': 'dataset', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagema
ker-us-east-1-390574811984/bias/balanced/womens_clothing_ecommerce_reviews_balanced.csv',
'LocalPath': '/opt/ml/processing/input/data', 'S3DataType': 'S3Prefix', 'S3InputMode': 'F
ile', 'S3DataDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}, {'Input
Name': 'analysis_config', 'AppManaged': False, 'S3Input': {'S3Uri': 's3://sagemaker-us-ea
st-1-390574811984/bias/generated_bias_report/balanced/analysis_config.json', 'LocalPath':
'/opt/ml/processing/input/config', 'S3DataType': 'S3Prefix', 'S3InputMode': 'File', 'S3Da
taDistributionType': 'FullyReplicated', 'S3CompressionType': 'None'}}]
Outputs:  [{'OutputName': 'analysis_result', 'AppManaged': False, 'S3Output': {'S3Uri': '
s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced', 'LocalPath':
'/opt/ml/processing/output', 'S3UploadMode': 'EndOfJob'}}]
```

In [35]:

```
run_balanced_bias_processing_job_name = clarify_processor_balanced.latest_job.job_name
print(run_balanced_bias_processing_job_name)
```

```
Clarify-Pretraining-Bias-2021-06-25-16-05-21-917
```

4.5. Run and review the Clarify processing job on the balanced dataset

Review the results of the run following the links:

In [36]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="https://console.aws.amazon.com/sagemaker/
home?region={}#/processing-jobs/{}">processing job</a></b>'.format(region, run_balanced_b
ias_processing_job_name)))
```

Review processing job

In [37]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="https://console.aws.amazon.com/cloudwatch
/home?region={}#logStream:group=/aws/sagemaker/ProcessingJobs;prefix={};streamFilter=type
LogStreamPrefix">CloudWatch logs</a> after about 5 minutes</b>'.format(region, run_balanc
ed_bias_processing_job_name)))
```

Review CloudWatch logs after about 5 minutes

In [38]:

```
running_processor = sagemaker.processing.ProcessingJob.from_processing_name(processing_jo
b_name=run_balanced_bias_processing_job_name,
                                                                             sagemaker_s
ession=sess)
```

This cell will take approximately 5-10 minutes to run.

In [39]:

```
%%time

running_processor.wait(logs=False)
```

```
.....................................................!CPU times: user 277 ms, s
ys: 33.8 ms, total: 311 ms
Wall time: 5min 16s
```

4.6. Analyze balanced bias report

List the files in the output path `bias_report_balanced_output_path`:

In [40]:

```
!aws s3 ls $bias_report_balanced_output_path/
```

```
2021-06-25 16:10:40      29889 analysis.json
2021-06-25 16:05:22        346 analysis_config.json
2021-06-25 16:10:40     394888 report.html
2021-06-25 16:10:40     129753 report.ipynb
2021-06-25 16:10:40     141422 report.pdf
```

Download generated bias report from S3 bucket:

In [41]:

```
!aws s3 cp --recursive $bias_report_balanced_output_path ./generated_bias_report/balance
d/
```

```
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced/analy
sis_config.json to generated_bias_report/balanced/analysis_config.json
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced/repor
t.ipynb to generated_bias_report/balanced/report.ipynb
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced/analy
sis.json to generated_bias_report/balanced/analysis.json
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced/repor
t.pdf to generated_bias_report/balanced/report.pdf
download: s3://sagemaker-us-east-1-390574811984/bias/generated_bias_report/balanced/repor
t.html to generated_bias_report/balanced/report.html
```

Review the downloaded bias report (in HTML format):

In [42]:

```
from IPython.core.display import display, HTML

display(HTML('<b>Review <a target="blank" href="./generated_bias_report/balanced/report.html">balanced bias report</a></b>'))
```

Review balanced bias report

In this run, you analyzed bias for `sentiment` relative to the `product_category` for the balanced data. Note that the Class Imbalance (CI) metric is equal across all product categories for the target label, `sentiment`. And Difference in Positive Proportions in Labels (DPL) metric values are zero.

Upload the notebook into S3 bucket for grading purposes.

Note: you may need to click on "Save" button before the upload.

In [44]:

```
!aws s3 cp ./C1_W2_Assignment.ipynb s3://$bucket/C1_W2_Assignment_Learner.ipynb
```

```
upload: ./C1_W2_Assignment.ipynb to s3://sagemaker-us-east-1-390574811984/C1_W2_Assignment_Learner.ipynb
```

Please go to the main lab window and click on `Submit` button (see the `Finish the lab` section of the instructions).

In [ ]:

```

```