

Final Project: Machine Learning Services  
Student Name: CHEN TZU-AN (Can also call me ANN)  
SMC ID: 1931977  
CHEN\_TZU-AN01@STUDENT.SMC.EDU

## AWS Machine Learning Report

Choose any project that you are interested in doing. The main thing is to learn and to have fun doing it. Come up with an innovative idea related to the course material that you are interested in related to Machine Learning on Amazon Web Services. Requirements:

- Select at least one Machine Learning related service on AWS. It could be any ML service we did or did not cover in this course.
- In this project, I used SageMaker to create a deployable model, provide bias detection and explainability to analyze data, and understand prediction outcomes from the model.
- Implement a working cloud architecture using the selected ML service.
- For this project, I downloaded and explored the South German Credit dataset to analyze credit risk. The data was preprocessed using sklearn, followed by training a Gradient Boosting Machine model with XGBoost. I then built an inference pipeline that combined the sklearn preprocessing model and the XGBoost model to preprocess input data and generate predictions for each instance. To ensure transparency and fairness, a SageMaker Clarify job was run to obtain Kernel SHAP values for the model on both training and test datasets and to provide bias metrics, including pre-training bias metrics on the data.
- The total time spent on the project - including research, design, configuring, coding, testing, redesign, etc., should be around 10 - 15 hours.

Submit:

- 3 full-page written report detailing the project.
- Include screenshots in the report Appendix

### [Appendix]

- South German Credit dataset source  
(<https://archive.ics.uci.edu/dataset/573/south+german+credit+update> )
- dataset source from Ulrike Grömping Beuth University of Applied Sciences Berlin  
(<https://prof.beuth-hochschule.de/groemping/>)
- A Unified Approach to Interpreting Model Predictions  
(<https://papers.nips.cc/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf> )
- Bias and Explainability-Tabular Data (<https://catalog.us-east-1.prod.workshops.aws/workshops/63069e26-921c-4ce1-9cc7-dd882ff62575/en-US/lab5-bais-and-explainablity/lab-1> )
- Host models along with pre-processing logic as serial inference pipeline behind one endpoint  
(<https://docs.aws.amazon.com/sagemaker/latest/dg/inference-pipelines.html>)
- aws/sagemaker-scikit-learn-container (<https://github.com/aws/sagemaker-scikit-learn-container>)
- aws/sagemaker-xgboost-container ([https://github.com/aws/sagemaker-xgboost-container/blob/master/src/sagemaker\\_xgboost\\_container/encoder.py](https://github.com/aws/sagemaker-xgboost-container/blob/master/src/sagemaker_xgboost_container/encoder.py) )
- Feature Attributions that Use Shapley Values  
(<https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-shapley-values.html>)
- Use SageMaker Clarify to explain and detect bias  
(<https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-configure-processing-jobs.html> )

# Credit risk prediction, explainability and bias detection with SageMaker

## Step 1:

### Initialize SageMaker

```
# cell 01
from io import StringIO
import os
import time
import sys
import IPython
from time import gmtime, strftime

import boto3
import numpy as np
import pandas as pd
import urllib

import sagemaker
from sagemaker.s3 import S3Uploader
from sagemaker.processing import ProcessingInput, ProcessingOutput
from sagemaker.sklearn.preprocessing import SKLearnProcessor
from sagemaker.inputs import TrainingInput
from sagemaker.xgboost import XGBoost
from sagemaker.s3 import S3Downloader
from sagemaker.s3 import S3Uploader
from sagemaker import Session
from sagemaker import get_execution_role
from sagemaker.xgboost import XGBoostModel
from sagemaker.sklearn import SKLearnModel
from sagemaker.pipeline import PipelineModel

session = Session()
bucket = session.default_bucket()
prefix = "sagemaker/sagemaker-clarify-credit-risk-model"
region = session.boto_region_name

# Define IAM role
role = get_execution_role()

[1] ... sagemaker.config INFO - Not applying SDK defaults from location: /etc/xdg/sagemaker/config.yaml
sagemaker.config INFO - Not applying SDK defaults from location: /home/sagemaker-user/.config/sagemaker/config.yaml
```

## Step 2:

### Download data

```
# cell 02
S3Downloader.download(
    "s3://sagemaker-sample-files/datasets/tabular/uci_statlog_german_credit_data/SouthGermanCredit.asc",
    "data",
)
[2] ... ['data/SouthGermanCredit.asc']
```

```
# cell 03
credit_columns = [
    "status",
    "duration",
    "credit_history",
    "purpose",
    "amount",
    "savings",
    "employment_duration",
    "installment_rate",
    "personal_status_sex",
    "other_debtors",
    "present_residence",
    "property",
    "age",
    "other_installment_plans",
    "housing",
    "number_credits",
    "job",
    "people_liable",
    "telephone",
    "foreign_worker",
    "credit_risk",
]
[3] Python

$ laufkont = status
1: no checking account
2: ... < 0 DM
3: 0<= ... < 200 DM
4: ... >= 200 DM / salary for at least 1 year

$ laufzeit = duration

$moral = credit_history
0 : delay in paying off in the past
1: critical account/other credits elsewhere
2 : no credits taken/all credits paid back duly 3 : existing credits paid back duly till now
4 : all credits at this bank paid back duly

$ verw = purpose
0 : others
1: car (new)
2 : car (used)
3 : furniture/equipment 4 : radio/television
5 : domestic appliances 6 : repairs
7 : education
```

```
信用評等可解釋性推論管道.ipynb > CS 79F > week 8 > credit_risk_explainability_inference_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 1. Overview
```

+ Markdown + 程式碼 ... 選取核心

```
8 : vacation  
9 : retraining  
10 : business
```

```
$hoehe = amount
```

```
$sparkont = savings
```

```
1 : unknown/no savings account 2 : ... < 100 DM  
3 : 100 <= ... < 500 DM  
4 : 500 <= ... < 1000 DM  
5 : ... >= 1000 DM
```

```
$beszeit = employment_duration
```

```
1 : unemployed  
2 : < 1 yr  
3 : 1 <= ... < 4 yrs 4 : 4 <= ... < 7 yrs 5 : >= 7 yrs
```

```
$rate = installment_rate
```

```
1 : >= 35  
2 : 25 <= ... < 35 3 : 20 <= ... < 25 4 : < 20
```

```
$famges = personal_status_sex
```

```
1 : male : divorced/separated  
2 : female : non-single or male : single 3 : male : married/widowed  
4 : female : single
```

```
$buerge = other_debtors
```

```
1 : none  
2 : co-applicant 3 : guarantor
```

```
$wohnzeit = present_residence
```

```
1 : < 1 yr  
2 : 1 <= ... < 4 yrs 3 : 4 <= ... < 7 yrs 4 : >= 7 yrs
```

```
$verm = property
```

```
1 : unknown / no property  
2 : car or other  
3 : building soc. savings agr./life insurance 4 : real estate
```

```
$alter = age
```

```
$weitkred = other_installment_plans
```

```
1 : bank  
2 : stores 3 : none
```

```
$wohn = housing
```

```
1 : for free 2 : rent
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb
- Search Bar:** 搜尋
- Toolbar:** Wed Jun 12 5:15 PM, various system icons.
- Code Cells:**
  - \$wohn = housing
  - 1:for free 2:rent  
3:own
  - \$bishkred = number\_credits
  - 1:1  
2:2-3 3:4-5 4:>= 6
  - \$beruf = job
  - 1:unemployed/unskilled - non-resident  
2:unskilled - resident  
3:skilled employee/official  
4:manager/self-empl./highly qualif. employee
  - \$pers = people\_liable
  - 1:3 or more 2:0 to 2
  - \$telef = telephone
  - 1:no  
2:yes (under customer name)
  - \$gastarb = foreign\_worker
  - 1:yes 2:no
  - \$kredit = credit\_risk
  - 0:bad 1:good
- Bottom Right:** 選取核心

### Step 3: Data inspection

Wed Jun 12 5:16 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

> CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 1. Overview

+ Markdown + 程式碼 ... 選取核心

```
# cell 04
training_data = pd.read_csv(
    "data/SouthGermanCredit.asc",
    names=credit_columns,
    header=0,
    sep=r" ",
    engine="python",
    na_values="?",
).dropna()

print(training_data.head())
[4] Python
```

... status duration credit\_history purpose amount savings \

	status	duration	credit_history	purpose	amount	savings	\
0	1	18	4	2	1049	1	
1	1	9	4	0	2799	1	
2	2	12	2	9	841	2	
3	1	12	4	0	2122	1	
4	1	12	4	0	2171	1	

employment\_duration installment\_rate personal\_status\_sex other\_debtors \

	employment_duration	installment_rate	personal_status_sex	other_debtors	\
0	2	4	2	1	
1	3	2	3	1	
2	4	2	2	1	
3	3	3	3	1	
4	3	4	3	1	

... property age other\_installment\_plans housing number\_credits job \

	property	age	other_installment_plans	housing	number_credits	job	\
0	...	2	21	3	1	1	3
1	...	1	36	3	1	2	3
2	...	1	23	3	1	1	2
3	...	1	39	3	1	2	2
4	...	2	38	1	2	2	2

people\_liable telephone foreign\_worker credit\_risk

	people_liable	telephone	foreign_worker	credit_risk
0	2	1	2	1
1	1	1	2	1
2	2	1	2	1
3	1	1	1	1
4	2	1	1	1

[5 rows x 21 columns]

#### Step 4:

Plotting histograms for the distribution of the different features to visualize the data.

Wed Jun 12 5:16 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

> CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 1. Overview

+ Markdown + 程式碼 ... 選取核心

```
# cell 05
%matplotlib inline
training_data["credit_risk"].value_counts().sort_values().plot(
    kind="bar", title="Counts of Target", rot=0
)
```

[5] Python

... <Axes: title={'center': 'Counts of Target'}, xlabel='credit\_risk'>

... Counts of Target

credit_risk	Counts
0	300
1	700

Step 5:

Create the raw training and test CSV files

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 1. Overview
- Code Cell 6:** Python code for preparing raw test data. It samples 10% of the training data, drops the "credit\_risk" column, and saves it as "test.csv". The columns listed are: status, duration, credit\_history, purpose, amount, savings, employment\_duration, installment\_rate, personal\_status\_sex, other\_debtors, present\_residence, property, age, other\_installment\_plans, housing, number\_credits, job, people\_liable, telephone, and foreign\_worker.
- Code Cell 7:** Python code for preparing raw training data. It saves the training data as "train.csv".

Step 6:  
Encode the training and test data for XGBoost models.

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 1. Overview
- Code Cell 7:** Python code for cell 07. It uses S3Uploader to upload the test file to the specified S3 bucket and prefix. The output shows the S3 URL: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/data/test/test.csv
- Code Cell 8:** Python code for cell 08. It uses S3Uploader to upload the train file to the specified S3 bucket and prefix. The output shows the S3 URL: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/data/train/train.csv

Step 7:  
Use SageMaker Processing jobs to perform the preprocessing on the raw data. Output a sklearn model that can be used for preprocessing inference requests.

A screenshot of a Jupyter Notebook interface. The title bar shows the file name "credit\_risk\_explainability\_inference\_pipelines.ipynb". The notebook navigation bar includes "Markdown", "程式碼", and "選取核心". The main content area displays Python code for defining a SKLearnProcessor:

```
# cell 09
sklearn_processor = SKLearnProcessor(
    role=role,
    base_job_name="sagemaker-clarify-credit-risk-processing-job",
    instance_type="ml.m5.large",
    instance_count=1,
    framework_version="0.20.0",
)
```

The code cell is labeled [9] and has a "Python" label in the bottom right corner.

## Step 8:

See the preprocessing script prepared to run in the processing job

A screenshot of a Jupyter Notebook interface, identical to the previous one but with a different code cell number. The title bar shows the file name "credit\_risk\_explainability\_inference\_pipelines.ipynb". The notebook navigation bar includes "Markdown", "程式碼", and "選取核心". The main content area displays Python code for a preprocessing script:

```
# cell 10
!pygmentize processing/preprocessor.py
```

The code cell is labeled [10] and has a "Python" label in the bottom right corner.

```
[10]
...
import argparse
import os
import warnings

import pandas as pd
import numpy as np
import tarfile
import sklearn
from sklearn.externals import joblib
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
from sklearn.compose import make_column_transformer

from sklearn.exceptions import DataConversionWarning

warnings.filterwarnings(action="ignore", category>DataConversionWarning)

columns = [
    "status",
    "duration",
    "credit_history",
    "purpose",
    "amount",
    "savings",
    "employment_duration",
...
    joblib.dump(featurizer_model, model_path)
    tar = tarfile.open(model_output_path, "w:gz")
    tar.add(model_path, arcname="model.joblib")
    tar.close()
```

An output message at the bottom of the cell says: "Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)..."

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Breadcrumbs:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > Train XGBoost
- Toolbar:** Includes icons for back, forward, search, and file operations.
- Cell Content:** Python code for setting paths and running a sklearn\_processor.

```
# cell 11
raw_data_path = "s3://{0}/{1}/data/train/".format(bucket, prefix)
train_data_path = "s3://{0}/{1}/data/preprocessed/train/".format(bucket, prefix)
val_data_path = "s3://{0}/{1}/data/preprocessed/val/".format(bucket, prefix)
model_path = "s3://{0}/{1}/sklearn/".format(bucket, prefix)

sklearn_processor.run(
    code="processing/preprocessor.py",
    inputs=[
        ProcessingInput(
            input_name="raw_data", source=raw_data_path, destination="/opt/ml/processing/input"
        )
    ],
    outputs=[
        ProcessingOutput(
            output_name="train_data", source="/opt/ml/processing/train", destination=train_data_path
        ),
        ProcessingOutput(
            output_name="val_data", source="/opt/ml/processing/val", destination=val_data_path
        ),
        ProcessingOutput(
            output_name="model", source="/opt/ml/processing/model", destination=model_path
        ),
    ],
    arguments=["--train-test-split-ratio", "0.2"],
    logs=False,
)
```

- Cell Number:** [12]
- Language:** Python
- Output:** INFO:sagemaker:Creating processing-job with name sagemaker-clarify-credit-risk-processin-2024-06-10-06-01-14-897

### Step 9:

Train XGBoost Model on the preprocessed data.

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > Train XGBoost
- Cell 12:** A code cell containing Python code for XGBoost training.
- Code Content:**

```
# cell 12
!pygmentize training/train_xgboost.py

[13] Python

...
import argparse
import json
import os
import random
import pandas as pd
import glob
import pickle as pkl

import xgboost

def parse_args():

    parser = argparse.ArgumentParser()

    parser.add_argument("--max_depth", type=int, default=5)
    parser.add_argument("--eta", type=float, default=0.05)
    parser.add_argument("--gamma", type=int, default=4)
    parser.add_argument("--min_child_weight", type=int, default=6)
    parser.add_argument("--silent", type=int, default=0)
    parser.add_argument("--objective", type=str, default="binary:logistic")
    parser.add_argument("--eval_metric", type=str, default="auc")
    parser.add_argument("--num_round", type=int, default=100)
    parser.add_argument("--subsample", type=float, default=0.8)
    parser.add_argument("--early_stopping_rounds", type=int, default=20)

    ...

if __name__ == "__main__":
    main()

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

## Step 10:

Set up XGBoost Estimator and pre-defined values for Hyperparameters for XGBoost algorithm

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Breadcrumbs:** CS 79F > week 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > Train XGBoost
- Toolbar:** Includes icons for back, forward, search, and file operations.
- Cell Content:** Python code for training an XGBoost model. The code defines hyperparameters, sets entry points, source directories, and output paths, and creates an estimator using the XGBoost class.

```
# cell 13
hyperparameters = {
    "max_depth": "5",
    "eta": "0.1",
    "gamma": "4",
    "min_child_weight": "6",
    "silent": "1",
    "objective": "binary:logistic",
    "num_round": "100",
    "subsample": "0.8",
    "eval_metric": "auc",
    "early_stopping_rounds": "20",
}

entry_point = "train_xgboost.py"
source_dir = "training/"
output_path = "s3://{}//{}/{}".format(bucket, prefix, "xgb_model")
code_location = "s3://{}//{}/code".format(bucket, prefix)

estimator = XGBoost(
    entry_point=entry_point,
    source_dir=source_dir,
    output_path=output_path,
    code_location=code_location,
    hyperparameters=hyperparameters,
    instance_type="ml.c5.xlarge",
    instance_count=1,
    framework_version="0.90-2",
    py_version="py3",
    role=role,
)
```

- Cell Number:** [14]
- Language:** Python

## Step 11:

Train the model

```
# cell 14
job_name = f"credit-risk-xgb-{strftime('%Y-%m-%d-%H-%M-%S', gmtime())}"

train_input = TrainingInput(
    "s3://{}-{}/data/preprocessed/train/'.format(bucket, prefix), content_type='csv"
)
val_input = TrainingInput(
    "s3://{}-{}/data/preprocessed/val/'.format(bucket, prefix), content_type='csv"
)

inputs = {"train": train_input, "validation": val_input}

estimator.fit(inputs, job_name=job_name)
```

[15] Python

```
... INFO:sagemaker:Creating training-job with name: credit-risk-xgb-2024-06-10-06-09-04
2024-06-10 06:09:04 Starting - Starting the training job...
2024-06-10 06:09:21 Starting - Preparing the instances for training...
2024-06-10 06:09:48 Downloading - Downloading input data...
2024-06-10 06:10:34 Training - Training image download completed. Training in progress..INFO:sagemaker-containers:Imported framework sagemaker
INFO:sagemaker-containers:No GPUs detected (normal if no gpus installed)
INFO:sagemaker_xgboost_container.training:Invoking user training script.
INFO:sagemaker-containers:Module train_xgboost does not provide a setup.py.
Generating setup.py
INFO:sagemaker-containers:Generating setup.cfg
INFO:sagemaker-containers:Generating MANIFEST.in
INFO:sagemaker-containers:Installing module with the following command:
/miniconda3/bin/python3 -m pip install .
Processing /opt/ml/code
Building wheels for collected packages: train-xgboost
  Building wheel for train-xgboost (setup.py): started
  Building wheel for train-xgboost (setup.py): finished with status 'done'
  Created wheel for train-xgboost: filename=train_xgboost-1.0.0-py2.py3-none-any.whl size=4571 sha256=6aebcc06a9bd7abb1ceb684853f629797a307fbe
  Stored in directory: /home/model-server/tmp/pip-ephem-wheel-cache-11b3yi4y/wheels/95/c1/85/65aaf48b35aba88c6e896d2fd04a4b69f1cee0d81ea32993c
Successfully built train-xgboost
Installing collected packages: train-xgboost
Successfully installed train-xgboost-1.0.0
INFO:sagemaker-containers:No GPUs detected (normal if no gpus installed)
INFO:sagemaker-containers:Invoking user script
Training Env:
{
...
2024-06-10 06:10:57 Uploading - Uploading generated training model
2024-06-10 06:10:57 Completed - Training job completed
Training seconds: 69
Billable seconds: 69
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

## Step 12:

Create two SageMaker Model objects, which associate the artifacts of training (to the Docker container used for inference, and define the model data location of SKlearn and XGBoost models.

```
# cell 15
preprocessor_model_data = "s3://{}-{}/{}".format(bucket, prefix, "sklearn") + "/model.tar.gz"

xgboost_model_data = (
    "s3://{}-{}/{}".format(bucket, prefix, "xgb_model", job_name) + "/output/model.tar.gz"
)
```

[16] Python

## Step 13:

Create a SageMaker SKlearn Model Object which will contain the location of the sklearn model data, custom inference code, and SKlearn version to use

The screenshot shows a Jupyter Notebook cell with the following code:

```
# cell 16
#pygmentize inference/sklearn/inference.py

[17]
... from __future__ import print_function

from io import StringIO
import os

import pandas as pd

from sklearn.externals import joblib

feature_columns_names = [
    "status",
    "duration",
    "credit_history",
    "purpose",
    "amount",
    "savings",
    "employment_duration",
    "installment_rate",
    "personal_status_sex",
    "other_debtors",
    "present_residence",
    ...
]

def model_fn(model_dir):
    preprocessor = joblib.load(os.path.join(model_dir, "model.joblib"))
    return preprocessor

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

The cell is labeled [17] and is set to run in Python.

## Step 14: Define the SKLearnModel Object

The screenshot shows a Jupyter Notebook cell with the following code:

```
# cell 17
sklearn_inference_code_location = "s3://{}//{}//{}//code".format(bucket, prefix, "sklearn")

sklearn_model = SKLearnModel(
    name="sklearn-model-{}".format(str(int(time.time()))),
    model_data=preprocessor_model_data,
    entry_point="inference.py",
    source_dir="inference/sklearn/",
    code_location=sklearn_inference_code_location,
    role=role,
    sagemaker_session=session,
    framework_version="0.20.0",
    py_version="py3",
)

[18]
```

The cell is labeled [18] and is set to run in Python.

## Step 15:

Create a XGBoost Model object and provide a custom inference script.

The screenshot shows a Jupyter Notebook interface with the title bar "credit\_risk\_explainability\_inference\_pipelines.ipynb". The notebook has two cells: Cell 18 and Cell 19. Cell 18 contains a command to run a Python script named "inference.py". Cell 19 contains the Python code for defining the XGBoost model and its inference function. The code uses the `xgboost` library and `sagemaker` containers to handle input data and return predictions.

```
# cell 18
!pygmentize inference/xgboost/inference.py

[19]
...
import pickle as pkl
import json
import numpy as np
import xgboost as xgb

from sagemaker_containers.beta.framework import content_types
from sagemaker_xgboost_container import encoder as xgb_encoders

def input_fn(input_data, content_type):
    if content_type == content_types.JSON:
        print("Received content type is json")
        print("input_data is", input_data)
        obj = json.loads(input_data)
        print("obj", obj)
        array = np.array(obj)
        return xgb.DMatrix(array)
    else:
        print("content type is not json")
        return xgb_encoders.decode(input_data, content_type)

def model_fn(model_dir):
    model_file = model_dir + "/model.bin"
    model = pkl.load(open(model_file, "rb"))
    return model
```

## Step 16:

Define the XGBoost model Object

The screenshot shows a Jupyter Notebook interface with the title bar "credit\_risk\_explainability\_inference\_pipelines.ipynb". The notebook has two cells: Cell 18 and Cell 20. Cell 18 contains a command to run a Python script named "inference.py". Cell 20 contains the Python code for defining the XGBoost model object. The code uses the `XGBModel` class from the `sagemaker\_xgboost\_container` module to define the model, specifying the model file, entry point, source directory, code location, framework version, Python version, role, and Sagemaker session.

```
# cell 19
xgboost_inference_code_location = "s3://{{}/{}/{}}/code".format(bucket, prefix, "xgb_model")

xgboost_model = XGBModel(
    name="xgb-model-{}".format(str(int(time.time()))),
    model_data=xgboost_model_data,
    entry_point="inference.py",
    source_dir="inference/xgboost/",
    code_location=xgboost_inference_code_location,
    framework_version="0.90-2",
    py_version="py3",
    role=role,
    sagemaker_session=session,
)
```

## Step 17:

Deploy the models in a pipeline by building a PipelineModel object and calling the deploy() method.

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 20
pipeline_model_name = "credit-risk-inference-pipeline-{0}".format(str(int(time.time())))

pipeline_model = PipelineModel(
    name=pipeline_model_name,
    role=role,
    models=[sklearn_model, xgboost_model],
    sagemaker_session=session,
)
```

[21] Python

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 21
pipeline_model.name
```

[22] Python

```
... 'credit-risk-inference-pipeline-1717999879'
```

Step 18:  
Deploy the model and test the inference pipeline.

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 22
endpoint_name = "credit-risk-pipeline-endpoint-{0}".format(str(int(time.time())))
print(endpoint_name)

pipeline_model.deploy(
    initial_instance_count=1, instance_type="ml.m5.xlarge", endpoint_name=endpoint_name
)
```

[23] Python

```
... credit-risk-pipeline-endpoint-1717999879
INFO:sagemaker.image_uris:Defaulting to only available Python version: py3
INFO:sagemaker:Creating model with name: credit-risk-inference-pipeline-1717999879
INFO:sagemaker:Creating endpoint-config with name credit-risk-pipeline-endpoint-1717999879
INFO:sagemaker:Creating endpoint with name credit-risk-pipeline-endpoint-1717999879
-----!
```

credit\_risk\_explainability\_inference\_pipelines.ipynb

# cell 23

```
test_dataset = S3Downloader.read_file(test_raw)

predictor = sagemaker.predictor.Predictor(
    endpoint_name,
    session,
    serializer=sagemaker.serializers.CSVSerializer(),
    deserializer=sagemaker.deserializers.CSVDeserializer(),
)
```

# cell 24

```
predictions
```

[24] [25]

```
[[0.87600094],  
[0.90517473],  
[0.64504521],  
[0.35770863],  
[0.78041881],  
[0.2566461],  
[0.88290461],  
[0.56853481],  
[0.84106421],  
[0.297732351],  
[0.63369951],  
[0.634264771],  
[0.94847021],  
[0.456270661],  
[0.84118441],  
[0.713510631],  
[0.89512041],  
[0.92473111],  
[0.70643521],  
[0.754563331],  
[0.95761411],  
[0.9622971],  
[0.63212781],  
[0.942648051],  
[0.159026681],  
...  
[0.80536321],  
[0.65274441],  
[0.844854351],  
[0.87570751],  
[0.95618621]]
```

Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

Step 19:  
SageMaker Clarify

```
# cell 25
from sagemaker import clarify

clarify_processor = clarify.SageMakerClarifyProcessor(
    role=role, instance_count=1, instance_type="ml.c4.xlarge", sagemaker_session=session
)

[26] INFO:sagemaker.image_uris:Defaulting to the only supported framework/algorithm version: 1.0.
INFO:sagemaker.image_uris:Ignoring unnecessary instance type: None.
```

Python

## Step 20:

To explain bad credit predictions, create a baseline for SHAP

```
# cell 26
# load the raw training data in a data frame
raw_train_df = pd.read_csv("train.csv", header=0, names=None, sep=",")

# drop the target column
baseline = raw_train_df.drop(["credit_risk"], axis=1).mode().iloc[0].values.astype("int").tolist()

print(baseline)

[27] [4, 24, 2, 3, 1258, 1, 3, 4, 3, 1, 4, 3, 27, 3, 2, 1, 3, 2, 1, 2]

# cell 27
# check baseline prediction E[f(x)]
pred_baseline = predictor.predict(baseline)
print(pred_baseline)

[28] [[0.78900737]]
```

Python

## Step 21:

Setup configurations (SHAPConfig, DataConfig, ModelConfig) to start the explainability analysis by Clarify

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Header:** Wed Jun 12 7:31PM
- Cells:** Three Python cells labeled [29], [30], and [82].
- Cell 29:** Configures SHAPConfig for explainability analysis.
- Cell 30:** Configures DataConfig for the explainability job.
- Cell 82:** Configures ModelConfig for the inference pipeline.

```
# cell 28
shap_config = clarify.SHAPConfig(
    baseline=[baseline],
    num_samples=2000, # num_samples are permutations from your features, so should be large enough as compared to number of input feature
    agg_method="mean_abs",
    use_logit=True,
) # we want the shap values to have log-odds units so that the equation 'shap values + expected probability = predicted probability' for

[29] Python

# cell 29
explainability_report_prefix = "{}/clarify-explainability".format(prefix)
explainability_output_path = "s3://{}{}".format(bucket, explainability_report_prefix)

explainability_data_config = clarify.DataConfig(
    s3_data_input_path=test_raw,
    s3_output_path=explainability_output_path,
    # label='credit_risk', # target column is not present in the test dataset
    headers=test_columns,
    dataset_type="text/csv",
)

[30] Python

# cell 30
model_config = clarify.ModelConfig(
    model_name=pipeline_model.name, # specify the inference pipeline model name
    instance_type="ml.c5.xlarge",
    instance_count=1,
    accept_type="text/csv",
)

[82] Python
```

## Step 22:

Run SageMaker Clarify Explainability job which spins up an ephemeral SageMaker endpoint, performs inference and calculate explanations on that endpoint

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Header:** Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model
- Cell Number:** [83]
- Language:** Python
- Code Content:**

```
# cell 31
clarify_processor.run_explainability(
    data_config=explainability_data_config,
    model_config=model_config,
    explainability_config=shap_config,
)
```
- Output Log:**

```
... INFO:sagemaker.clarify:Analysis Config: {'dataset_type': 'text/csv', 'headers': ['status', 'duration', 'credit_history', 'purpose', 'amount']}
INFO:sagemaker:Creating processing-job with name Clarify-Explainability-2024-06-10-07-02-20-554
.....WARNING:root:logging.conf not found when configuring logging, using default logging config
INFO:sagemaker-clarify-processing:Starting SageMaker Clarify Processing job
INFO:analyzer.data_loading.data_loader_util:Analysis config path: /opt/ml/processing/input/config/analysis_config.json
INFO:analyzer.data_loading.data_loader_util:Analysis result path: /opt/ml/processing/output
INFO:analyzer.data_loading.data_loader_util:This host is algo-1.
INFO:analyzer.data_loading.data_loader_util:This host is the leader.
INFO:analyzer.data_loading.data_loader_util:Number of hosts in the cluster is 1.
INFO:sagemaker-clarify-processing:Running Python / Pandas based analyzer.
INFO:analyzer.data_loading.data_loader_factory:Dataset type: text/csv uri: /opt/ml/processing/input/data
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated, df = df.append(df_tmp, ignore_index=True)
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated, df = df.append(df_tmp, ignore_index=True)
INFO:sagemaker-clarify-processing:Loading dataset...
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated, df = df.append(df_tmp, ignore_index=True)
INFO:sagemaker-clarify-processing:Loaded dataset. Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 100 entries, 0 to 99
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0   status            100 non-null    int64  
 1   duration          100 non-null    int64  
 2   credit_history    100 non-null    int64  
 ...
[NbConvertApp] Writing 511377 bytes to /opt/ml/processing/output/report.html
INFO:analyzer.utils.util:[wkhtmltopdf, '-q', '--enable-local-file-access', '/opt/ml/processing/output/report.html', '/opt/ml/processing/output/report.html']
INFO:analyzer.utils.system_util:exit_message: Completed: SageMaker XAI Analyzer ran successfully
---!
```
- Note:** Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...

## Step 23:

See the explainability report

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

:8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model

+ Markdown + 程式碼 ... 選取核心

```
[84] # cell 32
explainability_output_path
[84] Python
... 's3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-explainability'
```

```
[85] # cell 33
run_explainability_job_name = clarify_processor.latest_job.job_name
run_explainability_job_name
[85] Python
... 'Clarify-Explainability-2024-06-10-07-02-20-554'
```

```
[86] # cell 34
from IPython.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>
         region, run_explainability_job_name
    ')
)
[86] Python
...
```

```
[87] # cell 35
from IPython.display import display, HTML

display(
    HTML(
        '<b>Review <a target="blank" href="https://console.aws.amazon.com/cloudwatch/home?region={}#logStream:group=/aws/sagemaker/Process
         region, run_explainability_job_name
    ')
)
)
[87] Python
...
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows the file name "credit\_risk\_explainability\_inference\_pipelines.ipynb" and the date "Wed Jun 12 9:09 PM".
- Header:** Includes standard icons for file operations like back, forward, search, and help.
- Toolbar:** Shows the current cell number (8), the notebook title, and a progress bar indicating the status of the notebook.
- Code Cells:**
  - Cell 36:** Python code to display an S3 output URL. The code imports `IPython.display` and uses its `display` function to show an HTML string containing a link to an S3 bucket and prefix. The output is labeled "S3 Output Data After The Process".
  - Cell 37:** Python code to define the explainability output path. It sets `explainability\_output\_path` to the value "s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-explainability".
- Bottom Status:** Shows the cell index [88] and the language Python.

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Breadcrumbs:** illity\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 5. Amazon SageMaker Clarify > Detect data bias with A
- Toolbar:** Back, Forward, Search, etc.
- Cell 90 (Python):** !aws s3 ls \$explainability\_output\_path/  
Output:  
...  
PRE explanations\_shap/  
2024-06-10 07:16:02 1366 analysis.json  
2024-06-10 07:02:21 770 analysis\_config.json  
2024-06-10 07:16:02 511377 report.html  
2024-06-10 07:16:02 239053 report.ipynb  
2024-06-10 07:16:02 202543 report.pdf
- Cell 91 (Python):** !aws s3 cp --recursive \$explainability\_output\_path ./explainability\_report/  
Output:  
... download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-explainability/analysis.json to explainability/report/analysis.json  
download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-explainability/explanations\_shap/baselin...
- Text:** View the explainability pdf report below to see global explanations with SHAP for the model. The report also includes a SHAP summary plot for all individual instances in the dataset.
- Cell 92 (Python):** # cell 40  
from IPython.display import display, HTML  
display(HTML('<b>Review <a target="blank" href="./explainability\_report/report.html">Explainability Report</a></b>'))

## Step 24:

Clarify produces a CSV file which contains the SHAP value for each feature per prediction

Wed Jun 12 9:11 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

: 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model

+ Markdown + 程式碼 ... 選取核心

```
[93] # cell 41
from sagemaker.s3 import S3Downloader
import json
import io

# read the shap values
S3Downloader.download(s3_uri=explainability_output_path + "/explanations_shap", local_path="output")
shap_values_df = pd.read_csv("output/out.csv")
print(shap_values_df.shape)
```

[93] Python  
... (100, 20)

Note that by default SHAP explains classifier models in terms of their margin output, before the logistic link function. That means the units of SHAP output are log-odds units, so negative values imply probabilities of less than 0.5 meaning bad credit class (class 0).

A brief technical summary of prediction output before the logistic link function and SHAP values

$y = f(x)$  is the log-odd (logit) unit for the prediction output

$E(y)$  is the log-odd (logit) unit for the prediction on the input baseline

SHAP values are in log-odd units as well

The following is expected to hold true for every individual prediction :

```
sum(SHAP values) + E(y) == model_prediction_logit
logistic(model_prediction_logit) = model_prediction_probability
E(y) < 0 implies baseline probability less than 0.5 (bad credit baseline)
E(y) > 0 implies baseline probability greater than 0.5 (good credit baseline)
y < 0 implies predicted probability less than 0.5 (bad credit)
y > 0 implies predicted probability greater than 0.5 (good credit)
```

We can retrieve  $E(y)$ , the log-odd unit of the prediction for the baseline input

Wed Jun 12 9:16 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

: 8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model

+ Markdown + 程式碼 ... 選取核心

```
[94] ▷ # cell 42
# get the base expected value to be used to plot SHAP values
S3Downloader.download(s3_uri=explainability_output_path + "/analysis.json", local_path="output")

with open("output/analysis.json") as json_file:
    data = json.load(json_file)
    base_value = data["explanations"]["kernel_shap"]["label0"]["expected_value"]

print("E(y): ", base_value)
```

[94] Python  
... E(y): 1.3189524577840586

$E(y) > 0$  implies baseline probability greater than 0.5 (good credit baseline)

Step 25:

Create a dataframe containing the model predictions generated earlier during inference

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 43
from pandas import DataFrame

predictions_df = DataFrame(predictions, columns=["probability_score"])

predictions_df
```

[95]

Python

## Step 26:

Join the predictions, SHAP value and test data

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 44
# join the probability score and shap values together in a single data frame
predictions_df.reset_index(drop=True, inplace=True)
shap_values_df.reset_index(drop=True, inplace=True)
test_data.reset_index(drop=True, inplace=True)

prediction_shap_df = pd.concat([predictions_df, shap_values_df, test_data], axis=1)
prediction_shap_df["probability_score"] = pd.to_numeric(
    prediction_shap_df["probability_score"], downcast="float"
)

prediction_shap_df
```

[96]

Python

## Step 27:

Convert the probability score to a binary value(1/0), based on a threshold(0.5), where probability scores greater than 0.5 are positive outcomes (good credit) and lesser are negative outcomes (bad credit)

credit\_risk\_explainability\_inference\_pipelines.ipynb

```
# cell 45
# create a new column as 'Prediction' converting the probability score to either 1 or 0
prediction_shap_df.insert(
    0, "Prediction", (prediction_shap_df["probability_score"] > 0.5).astype(int)
)

prediction_shap_df
```

[97]

Python

## Step 28:

Filter for bad credit predictions only

```
# cell 46
bad_credit_outcomes_df = prediction_shap_df[prediction_shap_df.iloc[:, 0] == 0]
bad_credit_outcomes_df
```

[98]

Python

## Step 29:

Create SHAP plots to understand how much different features contributed to a specific negative outcome

```
# cell 47
!conda update -n base -c defaults conda -y
!pip uninstall numpy -y
!pip uninstall numba -y
!pip install numba
!conda install -c conda-forge shap -y
```

[99]

```
... Channels:
- defaults
- conda-forge
Platform: linux-64
Collecting package metadata (repodata.json): -
IOStream.flush timed out
Found existing installation: numpy 1.26.4
Uninstalling numpy-1.26.4:
  Successfully uninstalled numpy-1.26.4
Found existing installation: numba 0.59.1
Uninstalling numba-0.59.1:
  Successfully uninstalled numba-0.59.1
Collecting numba
  Using cached numba-0.59.1-cp310-cp310-manylinux2014_x86_64.manylinux_2_17_x86_64.whl.metadata (2.7 kB)
Requirement already satisfied: llvmlite<0.43,>=0.42.0dev0 in /opt/conda/lib/python3.10/site-packages (from numba) (0.42.0)
Collecting numpy<1.27,>=1.22 (from numba)
  Using cached numpy-1.26.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (61 kB)
Using cached numba-0.59.1-cp310-cp310-manylinux2014_x86_64.manylinux_2_17_x86_64.whl (3.7 MB)
Using cached numpy-1.26.4-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
Installing collected packages: numpy, numba
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the
autogluon-common 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-features 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-features 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires accelerate<0.22.0,>=0.21.0, but you have accelerate 0.23.0 which is incompatible.
autogluon-multimodal 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-multimodal 0.8.3 requires pytorch-lightning<1.10.0,>=1.9.0, but you have pytorch-lightning 2.0.9 which is incompatible.
autogluon-multimodal 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires torch<1.14,>=1.9, but you have torch 2.0.0.post104 which is incompatible.
...
Channels:
- conda-forge
- defaults
Platform: linux-64
Collecting package metadata (repodata.json): \
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings...
```

## Step 30:

Install open source SHAP library for more visualizations.

```
[8] credit_risk_explainability_inference_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model
```

+ Markdown + 程式碼 ... 選取核心

```
[103] Python
```

```
!pip install shap
```

```
Requirement already satisfied: shap in /opt/conda/lib/python3.10/site-packages (0.45.1)
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages (from shap) (1.26.4)
Requirement already satisfied: scipy in /opt/conda/lib/python3.10/site-packages (from shap) (1.11.4)
Requirement already satisfied: scikit-learn in /opt/conda/lib/python3.10/site-packages (from shap) (1.4.2)
Requirement already satisfied: pandas in /opt/conda/lib/python3.10/site-packages (from shap) (2.1.4)
Requirement already satisfied: tqdm>=4.27.0 in /opt/conda/lib/python3.10/site-packages (from shap) (4.66.4)
Requirement already satisfied: packaging>=20.9 in /opt/conda/lib/python3.10/site-packages (from shap) (23.2)
Requirement already satisfied: slicer==0.0.8 in /opt/conda/lib/python3.10/site-packages (from shap) (0.0.8)
Requirement already satisfied: numba in /opt/conda/lib/python3.10/site-packages (from shap) (0.59.1)
Requirement already satisfied: cloudpickle in /opt/conda/lib/python3.10/site-packages (from shap) (2.2.1)
Requirement already satisfied: llvmlite<0.43,>=0.42.0dev0 in /opt/conda/lib/python3.10/site-packages (from numba->shap) (0.42.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /opt/conda/lib/python3.10/site-packages (from pandas->shap) (2.9.0)
Requirement already satisfied: pytz>=2020.1 in /opt/conda/lib/python3.10/site-packages (from pandas->shap) (2023.3)
Requirement already satisfied: tzdata>=2022.1 in /opt/conda/lib/python3.10/site-packages (from pandas->shap) (2024.1)
Requirement already satisfied: joblib>=1.2.0 in /opt/conda/lib/python3.10/site-packages (from scikit-learn->shap) (1.4.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /opt/conda/lib/python3.10/site-packages (from scikit-learn->shap) (3.5.0)
Requirement already satisfied: six>=1.5 in /opt/conda/lib/python3.10/site-packages (from python-dateutil>=2.8.2->pandas->shap) (1.16.0)
```

```
[8] credit_risk_explainability_inference_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model
```

+ Markdown + 程式碼 ... 選取核心

```
[105] Python
```

```
!pip install numpy==1.25.2
```

```
... Collecting numpy==1.25.2
  Using cached numpy-1.25.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (5.6 kB)
  Using cached numpy-1.25.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the
autogluon-common 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-features 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-features 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires accelerate<0.22.0,>=0.21.0, but you have accelerate 0.23.0 which is incompatible.
autogluon-multimodal 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-multimodal 0.8.3 requires pytorch-lightning<1.10.0,>=1.9.0, but you have pytorch-lightning 2.0.9 which is incompatible.
autogluon-multimodal 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires torch<1.14,>=1.9, but you have torch 2.0.0.post104 which is incompatible.
autogluon-multimodal 0.8.3 requires torchmetrics<0.12.0,>=0.11.0, but you have torchmetrics 1.0.3 which is incompatible.
autogluon-multimodal 0.8.3 requires torchvision<0.15.0, but you have torchvision 0.15.2a0+ab7b3e6 which is incompatible.
autogluon-tabular 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-timeseries 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-timeseries 0.8.3 requires pytorch-lightning<1.10.0,>=1.7.4, but you have pytorch-lightning 2.0.9 which is incompatible.
autogluon-timeseries 0.8.3 requires torch<1.14,>=1.9, but you have torch 2.0.0.post104 which is incompatible.
hdijupyterutils 0.21.0 requires pandas<2.0.0,>=0.17.1, but you have pandas 2.1.4 which is incompatible.
sparkmagic 0.21.0 requires pandas<2.0.0,>=0.17.1, but you have pandas 2.1.4 which is incompatible.
Successfully installed numpy-1.25.2
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** credit\_risk\_explainability\_inference\_pipelines.ipynb
- Header:** Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model
- Cells:**
  - Cell 105:** Python code: 

```
!pip install numpy==1.25.2
```

 Output:

```
Collecting numpy==1.25.2
  Using cached numpy-1.25.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (5.6 kB)
Using cached numpy-1.25.2-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (18.2 MB)
Installing collected packages: numpy
  Attempting uninstall: numpy
    Found existing installation: numpy 1.26.4
    Uninstalling numpy-1.26.4:
      Successfully uninstalled numpy-1.26.4
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the autogluon-common 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-core 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-features 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-features 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires accelerate<0.22.0,>=0.21.0, but you have accelerate 0.23.0 which is incompatible.
autogluon-multimodal 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-multimodal 0.8.3 requires pytorch-lightning<1.10.0,>=1.9.0, but you have pytorch-lightning 2.0.9 which is incompatible.
autogluon-multimodal 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-multimodal 0.8.3 requires torch<1.14,>=1.9, but you have torch 2.0.0.post104 which is incompatible.
autogluon-multimodal 0.8.3 requires torchmetrics<0.12.0,>=0.11.0, but you have torchmetrics 1.0.3 which is incompatible.
autogluon-multimodal 0.8.3 requires torchvision<0.15.0, but you have torchvision 0.15.2a0+ab7b3e6 which is incompatible.
autogluon-tabular 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-tabular 0.8.3 requires scikit-learn<1.4.1,>=1.1, but you have scikit-learn 1.4.2 which is incompatible.
autogluon-timeseries 0.8.3 requires pandas<1.6,>=1.4.1, but you have pandas 2.1.4 which is incompatible.
autogluon-timeseries 0.8.3 requires pytorch-lightning<1.10.0,>=1.7.4, but you have pytorch-lightning 2.0.9 which is incompatible.
autogluon-timeseries 0.8.3 requires torch<1.14,>=1.9, but you have torch 2.0.0.post104 which is incompatible.
hdijupyterutils 0.21.0 requires pandas<2.0.0,>=0.17.1, but you have pandas 2.1.4 which is incompatible.
sparkmagic 0.21.0 requires pandas<2.0.0,>=0.17.1, but you have pandas 2.1.4 which is incompatible.
Successfully installed numpy-1.25.2
```
  - Cell 106:** Python code: 

```
# cell 48
import shap
```
- Toolbar:** Includes back, forward, search, and file operations.
- Header Buttons:** Includes a gear icon for settings.
- Header Text:** Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model
- Header Buttons:** Includes a gear icon for settings.

### Step 31:

SHAP explanation plot for a single bad credit ensemble prediction instance.

Wed Jun 12 9:49 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

:8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model

+ Markdown + 程式碼 ... 選取核心

D # cell 49

```
import matplotlib.pyplot as plt

min_index = prediction_shap_df["probability_score"].idxmin()
print(min_index)
print("mean probability of dataset")
print(prediction_shap_df[["probability_score"]].mean())
print("individual probability")
print(prediction_shap_df.iloc[min_index, 1])
print("sum of shap values")
print(prediction_shap_df.iloc[min_index, 2:22].sum())
print("base value from analysis.json")
print(base_value)
```

[107] Python

... 24

```
mean probability of dataset
probability_score      0.710513
dtype: float32
individual probability
0.15902668
sum of shap values
-2.9844404062301186
base value from analysis.json
1.3189524577840586
```

### Step 32:

Example 'bad credit' prediction SHAP values, the input features (status = 1) , (purpose = 0) and (personal\_status\_sex = 2) are the top 3 features driving the negative decision.

Wed Jun 12 9:51 PM

credit\_risk\_explainability\_inference\_pipelines.ipynb ×

:8 > credit\_risk\_explainability\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 4. Create SageMaker Model

+ Markdown + 程式碼 ... 選取核心

D # cell 50

```
import inspect
import shap.plots._waterfall
source = inspect.getsource(shap.plots._waterfall)
new_source = source.replace("as pl", "as plt")
exec(new_source, shap.plots._waterfall.__dict__)

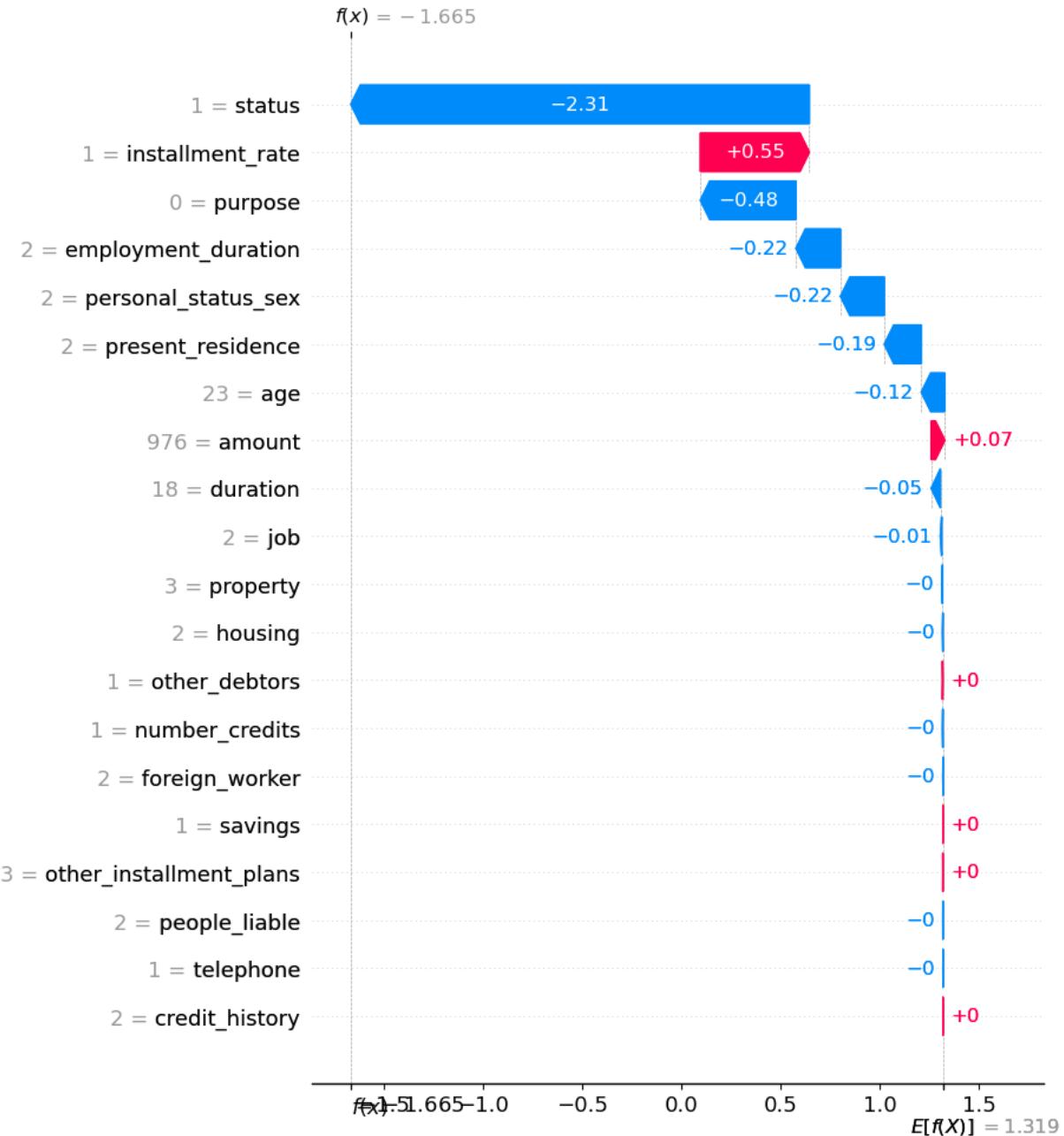
explanation_obj = shap.Explanation(
    values=prediction_shap_df.iloc[min_index, 2:22].to_numpy(),
    base_values=base_value,
    data=test_data.iloc[min_index].to_numpy(),
    feature_names=test_data.columns,
)
shap.plots.waterfall(shap_values=explanation_obj, max_display=20, show=False)
```

[109] Python

... <Axes: >

D plt.show()

[110] Python



Step 33:

Specify where to find the input dataset, where to store the output, the target column (label), the header names, and the dataset type.

```
# cell 51
bias_report_prefix = "{}/clarify-bias".format( prefix)
bias_report_output_path = "s3://{}{}".format(bucket,bias_report_prefix)
bias_data_config = clarify.DataConfig(
    s3_data_input_path=train_raw,
    s3_output_path=bias_report_output_path,
    label="credit_risk",
    headers=training_data.columns.to_list(),
    dataset_type="text/csv",
)
predictions_config = clarify.ModelPredictedLabelConfig(label=None, probability=0)
```

[111] Python

Step 34:

The group 'personal\_status\_sex' is used to form subgroups for the measurement of Conditional Demographic Disparity (CDD) metric only.

```
# cell 52
bias_config = clarify.BiasConfig(
    label_values_or_threshold=[1],
    facet_name="age",
    facet_values_or_threshold=[40],
    group_name="personal_status_sex",
)
```

[112] Python

```
# cell 53
clarify_processor.run_bias(
    data_config=bias_data_config,
    bias_config=bias_config,
    model_config=model_config,
    model_predicted_label_config=predictions_config,
    pre_training_methods="all",
    post_training_methods="all",
)

[113] Python
...
INFO:sagemaker.clarify:Analysis Config: {'dataset_type': 'text/csv', 'headers': ['status', 'duration', 'credit_history', 'purpose', 'amount', '']
INFO:sagemaker:Creating processing-job with name Clarify-Bias-2024-06-10-07-36-16-236
..... WARNING:root:logging.conf not found when configuring logging, using default logging config
INFO:sagemaker-clarify-processing:Starting SageMaker Clarify Processing job
INFO:analyzer.data_loading.data_loader_util:Analysis config path: /opt/ml/processing/input/config/analysis_config.json
INFO:analyzer.data_loading.data_loader_util:Analysis result path: /opt/ml/processing/output
INFO:analyzer.data_loading.data_loader_util:This host is algo-1.
INFO:analyzer.data_loading.data_loader_util:This host is the leader.
INFO:analyzer.data_loading.data_loader_util:Number of hosts in the cluster is 1.
INFO:sagemaker-clarify-processing:Running Python / Pandas based analyzer.
INFO:analyzer.data_loading.data_loader_factory:Dataset type: text/csv uri: /opt/ml/processing/input/data
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated; df = df.append(df_tmp, ignore_index=True)
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated; df = df.append(df_tmp, ignore_index=True)
INFO:sagemaker-clarify-processing>Loading dataset...
/usr/local/lib/python3.9/site-packages/analyzer/data_loading/data_readers/csv_data_reader.py:58: FutureWarning: The frame.append method is deprecated; df = df.append(df_tmp, ignore_index=True)
INFO:sagemaker-clarify-processing:Loaded dataset. Dataset info:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 20 columns):
 #   Column           Non-Null Count  Dtype  
 ---  -- 
 0   status            1000 non-null   int64  
 1   duration          1000 non-null   int64  
 2   credit_history    1000 non-null   int64  
 ...
INFO:analyzer.utils.system_util:exit_message: Completed: SageMaker XAI Analyzer ran successfully
INFO:py4j.clientserver:Closing down clientserver connection
---!
```

Output is truncated. View as a [scrollable element](#) or open in a [text editor](#). Adjust cell output [settings](#)...

## Step 35: Viewing the Bias detection Report

信用\_inference\_pipelines.ipynb > Credit risk prediction, explainability and bias detection with Amazon SageMaker > 5. Amazon SageMaker Clarify > Detect data bias with A

+ Markdown + 程式碼 ... 選取核心

```
[114] # cell 54
bias_report_output_path
[114] Python
...
's3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias'

[115] # cell 55
run_post_training_bias_processing_job_name = clarify_processor.latest_job.job_name
run_post_training_bias_processing_job_name
[115] Python
...
'Clarify-Bias-2024-06-10-07-36-16-236'

[116] # cell 56
from IPython.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>' +
        ' region, run_post_training_bias_processing_job_name'
    )
)
[116] Python
...
[117] # cell 57
from IPython.display import display, HTML
display(
    HTML(
        '<b>Review <a target="blank" href="https://console.aws.amazon.com/cloudwatch/home?region={}#logStream:group=/aws/sagemaker/Process' +
        ' region, run_post_training_bias_processing_job_name'
    )
)
[117] Python
...
```

# cell 58  
from IPython.display import display, HTML  
  
display(  
 HTML(  
 '<b>Review <a target="blank" href="https://s3.console.aws.amazon.com/s3/buckets/{}?prefix={}"/>S3 Output Data</a> After The Proces  
 bucket, bias\_report\_prefix  
 )  
)

[118] ... Python

## Step 36: Download Report from S3

# cell 59  
!aws s3 ls \$bias\_report\_output\_path/  
[119] ... 2024-06-10 07:49:38 5554 analysis.json  
2024-06-10 07:36:17 861 analysis\_config.json  
2024-06-10 07:49:38 777289 report.html  
2024-06-10 07:49:38 505476 report.ipynb  
2024-06-10 07:49:38 482383 report.pdf

# cell 60  
bias\_report\_output\_path  
[120] ... 's3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias'

# cell 61  
!aws s3 cp --recursive \$bias\_report\_output\_path ./generated\_bias\_report/  
[121] ... download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias/analysis\_config.json to generated\_bias\_report/analysis\_config.json  
download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias/analysis.json to generated\_bias\_report/analysis.json  
download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias/report.ipynb to generated\_bias\_report/report.ipynb  
download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias/report.html to generated\_bias\_report/report.html  
download: s3://sagemaker-us-east-1-533267098338/sagemaker/sagemaker-clarify-credit-risk-model/clarify-bias/report.pdf to generated\_bias\_report/report.pdf

## Step 37: View the bias report pdf that contains the pre-training bias and post-training bias metrics.

```
# cell 62
from IPython.display import display, HTML

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

[122] Python

### Step 38:

#### View Bias Report in Studio

```
# cell 63
S3Downloader.download(s3_uri=bias_report_output_path + "/analysis.json", local_path="output")

with open("output/analysis.json") as json_file:
    data = json.load(json_file)
    print("pre-training bias metrics")
    class_imbalance = data["pre_training_bias_metrics"]["facets"]["age"][0]["metrics"][1]["value"]
    print("class imbalance: ", class_imbalance)
    DPL = data["pre_training_bias_metrics"]["facets"]["age"][0]["metrics"][2]["value"]
    print("DPL: ", DPL)
    print("\n")
    print("post training bias metrics")
    DPPL = data["post_training_bias_metrics"]["facets"]["age"][0]["metrics"][6]["value"]
    print("DPPL: ", DPPL)
    DI = data["post_training_bias_metrics"]["facets"]["age"][0]["metrics"][5]["value"]
    print("DI: ", DI)
```

[123] Python

```
... pre-training bias metrics
class imbalance: 0.456
DPL: -0.04848093083387206

post training bias metrics
DPPL: -0.03462346477052358
DI: 1.0436843714955653
```

### Step 39:

#### Clean Up

```
# cell 64
session.delete_endpoint(endpoint_name)
```

Python

```
# cell 65
session.delete_model(pipeline_model.name)
```

Python

# Analysis Report

## Global dataset report

This report is the output of the Amazon SageMaker Clarify analysis. The report is split into following parts:

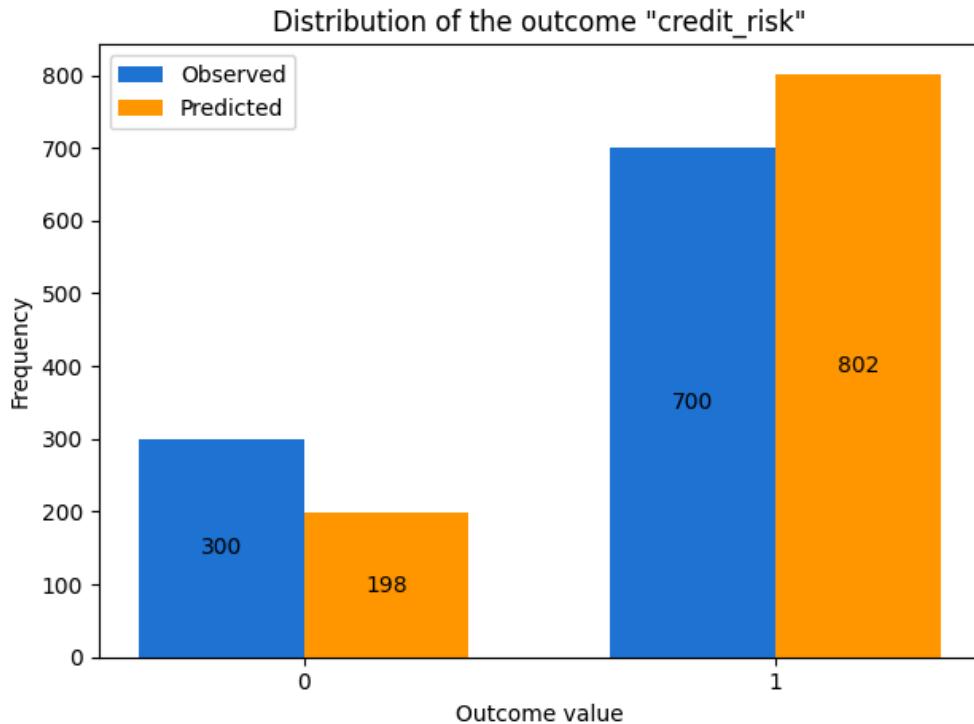
1. Analysis configuration
2. High level model performance
3. Pretraining bias metrics
4. Posttraining bias metrics

## Analysis Configuration

Bias analysis requires you to configure the outcome label column, the facet and optionally a group variable. Generating explanations requires you to configure the outcome label. You configured the analysis with the following variables. The complete analysis configuration is appended at the end.

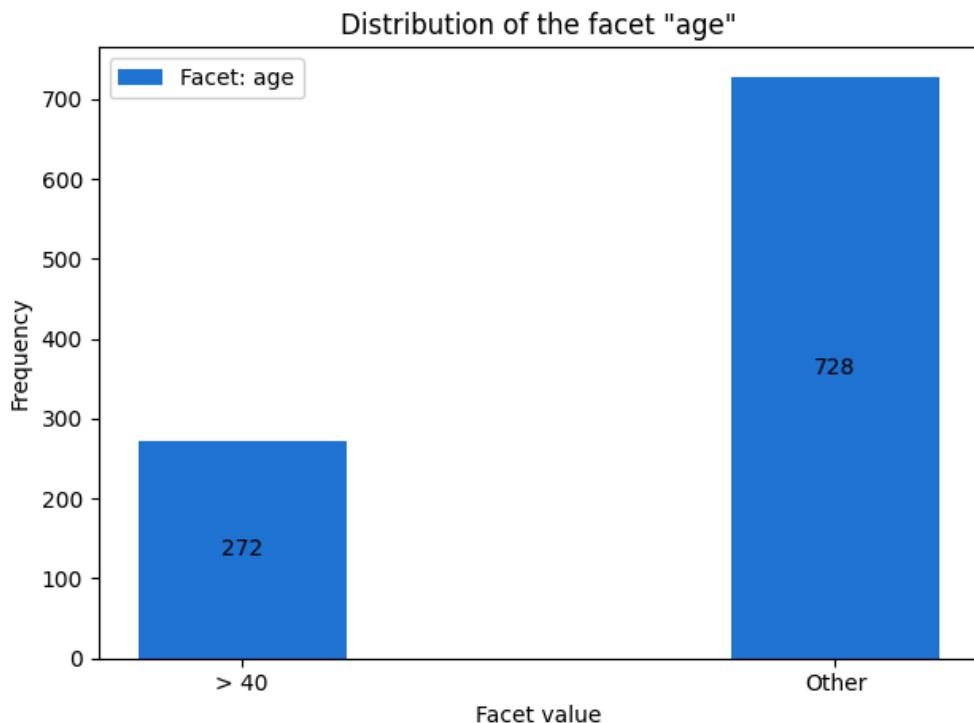
**Outcome label:** You chose the column `credit_risk` in the input data as the outcome label. Bias metric computation requires designating the positive outcome. You chose `credit_risk = 1` as the positive outcome. `credit_risk` consisted of values `[0, 1]`.

The figure below shows the distribution of values of `credit_risk` .



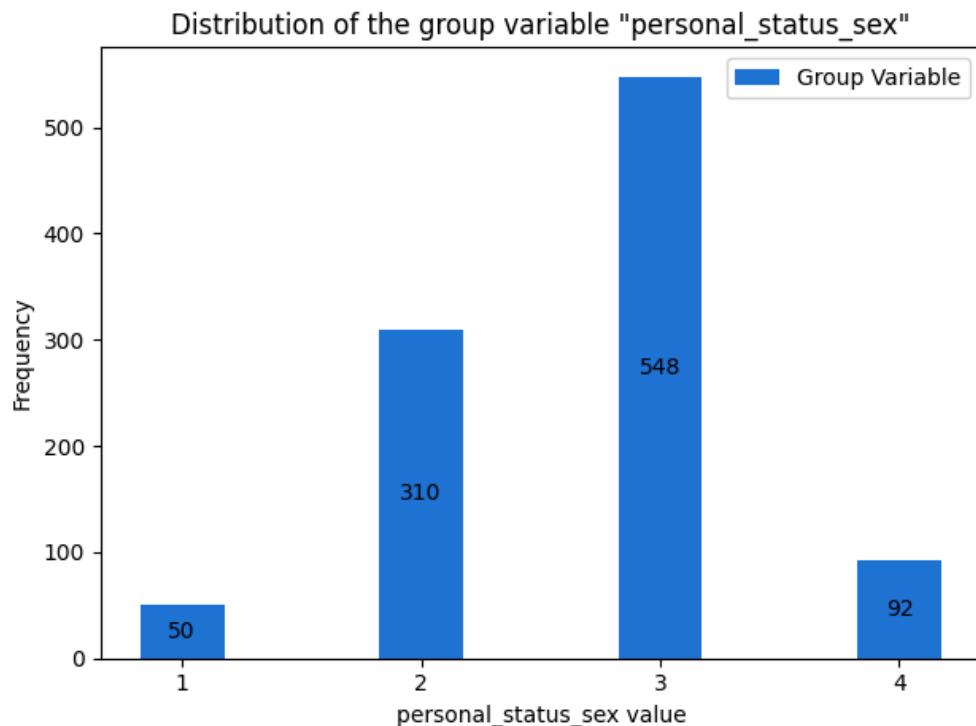
**Facet:** You chose the column `age` in the input data as the facet. `age` varied between `19.00` and `75.00` . Bias metrics were computed by comparing the inputs `age > 40` with all other inputs.

The figure below shows the distribution of values of `age` .



**Group variable:** Some bias metrics require an additional grouping variable. You chose `personal_status_sex` as the grouping variable.

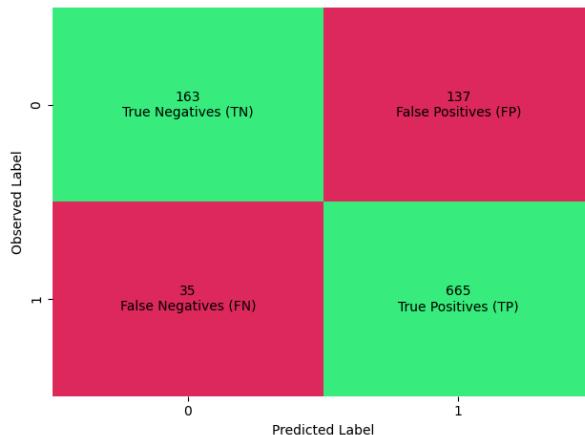
The figure below shows the distribution of values of `personal_status_sex`.



# High level model performance

Input data points can be divided into different categories based on their observed and predicted label. For instance, a **False Negative (FN)** is an input with a positive observed label (`credit_risk = 1`) but negative predicted label (`credit_risk != 1`). A **True Negative (TN)** is an input whose observed and predicted labels are both negative. **True Positives (TP)** and **False Positives (FP)** are defined similarly.

Based on the model predictions, the inputs can be divided into different categories as:

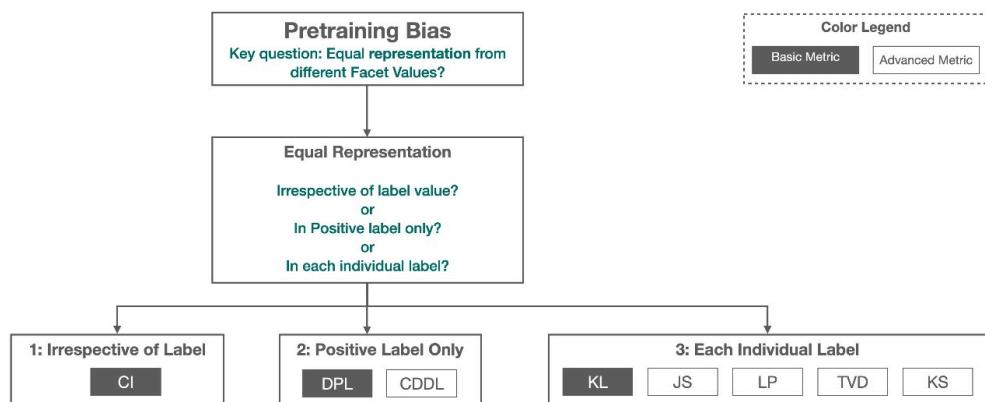


Here are metrics showing the model performance.

Metric	Description	Value
Accuracy	Proportion of inputs assigned the correct predicted label by the model.	0.828
Proportion of Positive Predictions in Labels	Proportion of input assigned in positive predicted label.	0.802
Proportion of Negative Predictions in Labels	Proportion of input assigned the negative predicted label.	0.198
True Positive Rate / Recall	Proportion of inputs with positive observed label correctly assigned the positive predicted label.	0.950
True Negative Rate / Specificity	Proportion of inputs with negative observed label correctly assigned the negative predicted label.	0.543
Acceptance Rate / Precision	Proportion of inputs with positive predicted label that actually have a positive observed label.	0.829
Rejection Rate	Proportion of inputs with negative predicted label that actually have a negative observed label.	0.823
Conditional Acceptance	Ratio between the positive observed labels and positive predicted labels.	0.873
Conditional Rejection	Ratio between the negative observed labels and negative predicted labels.	1.515
F1 Score	Harmonic mean of precision and recall.	0.885

# Pre-training Bias Metrics

Pretraining bias metrics measure imbalances in facet value representation in the training data. Imbalances can be measured across different dimensions. For instance, you could focus imbalances within the inputs with positive observed label only. The figure below shows how different pretraining bias metrics focus on different dimensions. For a detailed description of these dimensions, see [Learn How Amazon SageMaker Clarify Helps Detect Bias](https://aws.amazon.com/blogs/machine-learning/learn-how-amazon-sagemaker-clarify-helps-detect-bias/) (<https://aws.amazon.com/blogs/machine-learning/learn-how-amazon-sagemaker-clarify-helps-detect-bias/>).

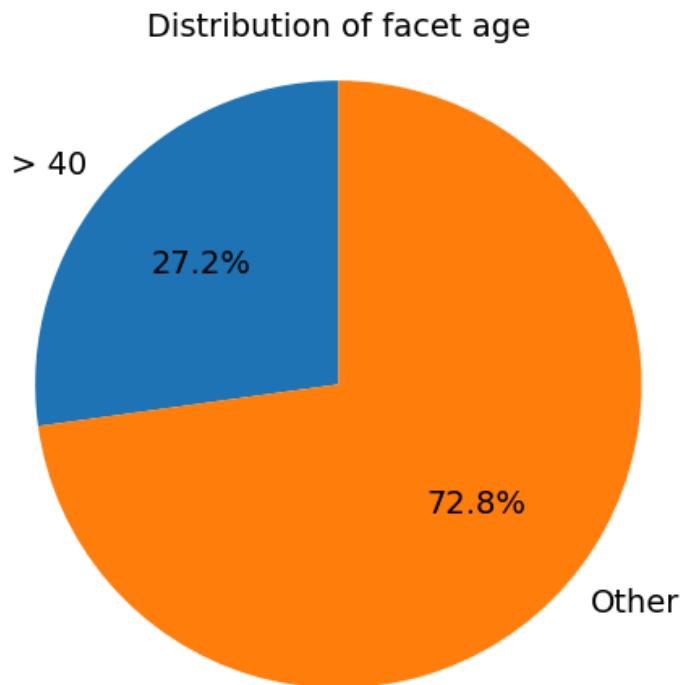


The metric values along with an informal description of what they mean are shown below. For mathematical formulas and examples, see the [Measure Pretraining Bias] (<https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-measure-data-bias.html>) section of the AWS documentation.

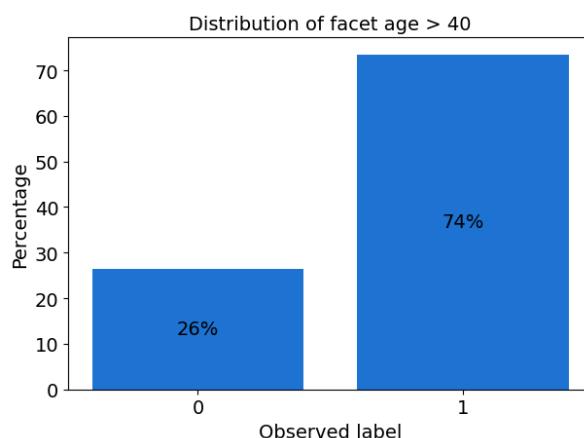
We computed the bias metrics for the label `credit_risk` using label value(s)/threshold `credit_risk = 1` for the following facets:

- Facet column: `age`

The pie chart shows the distribution of facet column `age` in your data.



The bar plot(s) below show the distribution of facet column `age` in your data.

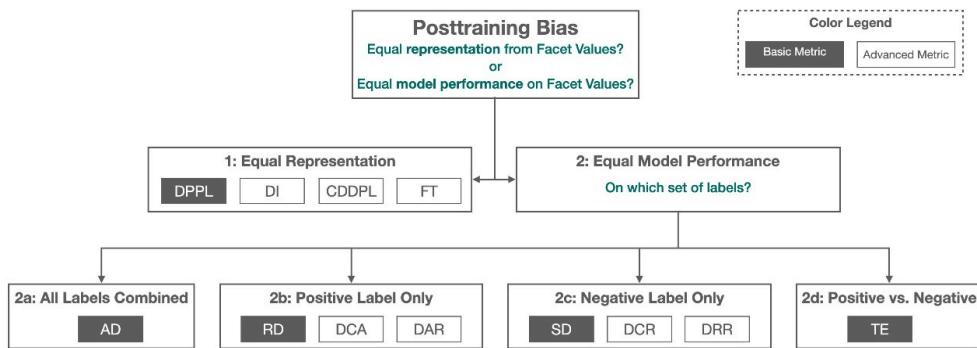


Facet Value(s)/Threshold: `age > 40`

Metric	Description	Value
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-cddl.html">Conditional Demographic Disparity in Labels (CDDL)</a>	Measures maximum divergence between the observed label distributions for facet values age > 40 and rest of the inputs in the dataset.	-0.035
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-bias-metric-class-imbalance.html">Class Imbalance (CI)</a>	Measures the imbalance in the number of inputs with facet values age > 40 and rest of the inputs.	0.456
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-kl-divergence.html">Difference in Proportions of Labels (DPL)</a>	Measures the imbalance of positive observed labels between facet values age > 40 and rest of the inputs.	-0.048
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-jensen-shannon-divergence.html">Jensen-Shannon Divergence (JS)</a>	Measures how much the observed label distributions of facet values age > 40 and rest of the inputs diverge from each other entropically.	0.001
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-kl.html">Kullback-Leibler Divergence (KL)</a>	Measures how much the observed label distributions of facet values age > 40 and rest of the inputs diverge from each other entropically.	0.006
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-kolmogorov-smirnov.html">Kolmogorov-Smirnov (KS)</a>	Measures maximum divergence between the observed label distributions for facet values age > 40 and rest of the inputs in the dataset.	0.048
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-lp-norm.html">L<sub>p</sub>-norm (LP)</a>	Measures a p-norm difference between the observed label distributions associated with facet values age > 40 rest of the inputs in the dataset.	0.069
<a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-data-bias-metric-total-variation-distance.html">Total Variation Distance (TVD)</a>	Measures half of the L1-norm difference between the observed label distributions associated with facet values age > 40 and rest of the inputs in the dataset.	0.048

# Post-training Bias Metrics

Posttraining bias metrics measure imbalances in model predictions across different inputs. The figure below shows how different posttraining metrics target different types of imbalances over inputs. For a detailed description of these types, see [Learn How Amazon SageMaker Clarify Helps Detect Bias](https://aws.amazon.com/blogs/machine-learning/learn-how-amazon-sagemaker-clarify-helps-detect-bias/) (<https://aws.amazon.com/blogs/machine-learning/learn-how-amazon-sagemaker-clarify-helps-detect-bias/>).

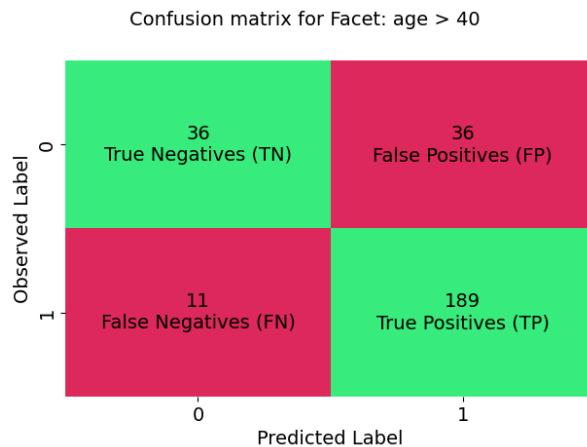


Bias can also result from imbalances in the model outcomes even when the facet value is not considered. The metric computing these imbalances is GE. The metric values along with an informal description of what they mean are shown below. For mathematical formulas and examples, see the [Measure Posttraining Data and Model Bias] (<https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-measure-post-training-bias.html>) section of the AWS documentation.

We computed the bias metrics for the label `credit_risk` using label value(s)/threshold `credit_risk = 1` for the following facets:

- Facet column: **age**

Facet Value(s)/Threshold: `age > 40`



Metric	Description	Value
<u>Accuracy Difference (AD)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ad.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ad.html</a> )	Measures the difference between the prediction accuracy for facet values age > 40 and rest of the inputs.	0.001
<u>Conditional Demographic Disparity in Predicted Labels (CDDPL)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dppl.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dppl.html</a> )	Measures the disparity of predicted labels between facet values age > 40 and rest of the inputs as a whole, but also by subgroups dictated by Age.	-0.020
<u>Difference in Acceptance Rates (DAR)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dcr.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dcr.html</a> )	Measures the difference in the ratios of the observed positive outcomes (TP) to the predicted positives (TP + FP) between facet values age > 40 and rest of the inputs.	-0.015
<u>Difference in Conditional Acceptance (DCAcc)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dca.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dca.html</a> )	Compares the observed labels to the labels predicted by the model. Assesses whether this is the same across facet values age > 40 and rest of the inputs for predicted positive outcomes (acceptances).	-0.022
<u>Difference in Conditional Rejection (DCR)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dcr.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dcr.html</a> )	Compares the observed labels to the labels predicted by the model and assesses whether this is the same across facet values age > 40 and rest of the inputs for negative outcomes (rejections).	0.022
<u>Disparate Impact (DI)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-di.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-di.html</a> )	Measures the ratio of proportions of the predicted labels for facet values age > 40 and rest of the inputs.	1.044
<u>Difference in Positive Proportions in Predicted Labels (DPPL)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dppl.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-dppl.html</a> )	Measures the difference in the proportion of positive predictions between facet values age > 40 and rest of the inputs.	-0.035
<u>Difference in Rejection Rates (DRR)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-drr.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-drr.html</a> )	Measures the difference in the ratios of the observed negative outcomes (TN) to the predicted negatives (TN + FN) between facet values age > 40 and rest of the inputs.	-0.075
<u>Counterfactual Fliptest (FT)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ft.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ft.html</a> )	Examines each input with facet value age > 40 and assesses whether similar members from rest of the inputs have different model predictions.	-0.107
<u>Generalized entropy (GE)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ge.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-ge.html</a> )	Measures the inequality in benefits b assigned to each input by the model predictions.	0.067
<u>Recall Difference (RD)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-rd.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-rd.html</a> )	Measures the difference between the recall, aka true positive rate, of the model for facet values age > 40 and rest of the inputs.	0.007
<u>Specificity difference (SD)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-sd.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-sd.html</a> )	Measures the difference between the specificity, aka true negative rate, of the model for facet values age > 40 and rest of the inputs.	-0.057
<u>Treatment Equality (TE)</u> ( <a href="https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-te.html">https://docs.aws.amazon.com/sagemaker/latest/dg/clarify-post-training-bias-metric-te.html</a> )	Measures the difference in the ratio of false positives to false negatives between facet values age > 40 and rest of the inputs.	0.068

## Appendix: Analysis Configuration Parameters

```
{  
    "dataset_type": "text/csv",  
    "headers": [  
        "status",  
        "duration",  
        "credit_history",  
        "purpose",  
        "amount",  
        "savings",  
        "employment_duration",  
        "installment_rate",  
        "personal_status_sex",  
        "other_debtors",  
        "present_residence",  
        "property",  
        "age",  
        "other_installment_plans",  
        "housing",  
        "number_credits",  
        "job",  
        "people_liable",  
        "telephone",  
        "foreign_worker",  
        "credit_risk"  
    ],  
    "label": "credit_risk",  
    "label_values_or_threshold": [  
        1  
    ],  
    "facet": [  
        {  
            "name_or_index": "age",  
            "value_or_threshold": [  
                40  
            ]  
        }  
    ],  
    "group_variable": "personal_status_sex",  
    "methods": {  
        "report": {  
            "name": "report",  
            "title": "Analysis Report"  
        },  
        "pre_training_bias": {  
            "methods": "all"  
        },  
        "post_training_bias": {  
            "methods": "all"  
        }  
    },  
    "predictor": {
```

```
"model_name": "credit-risk-inference-pipeline-1717999879",
"instance_type": "ml.c5.xlarge",
"initial_instance_count": 1,
"accept_type": "text/csv",
"probability": 0
}
}
```