

Configuration Manual

MSc Research Project

Data Analytics

Shreyal Kulaye

Student ID: 22155791

School of Computing

National College of Ireland

Supervisor: Abdul Qayum

National College of Ireland
MSc Project Submission Sheet



School of Computing

Student Name: Shreyal Ashok Kulaye

Student ID: 22155791

Programme : Research Project

Year : 2023

Module: Msc Data Analytics

Supervisor: Abdul Qayum
Submission Due Date:

Project Title: AirOpsAI: Unleashing the Power of AI and Data Science to revolutionise the Airlines Operations
Word Count: 1859 (Excluding references) **Page Count:** 16

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Shreyal Kulaye

Date: 14 December 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission , to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project , both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on the computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Shreyal Kulaye
Student ID: 22155791

Introduction

The Configuration Manual, a comprehensive set of instructions for configuring your system for AI and machine learning applications. This book offers vital information on what you need in terms of hardware and software, whether you are working on daily duties or special initiatives like enhancing aircraft operations. It guarantees a well configured environment for effective development, testing, and deployment, spanning from data science to deep learning and neural networks. By adhering to these rules, you will facilitate your code's smooth functioning and advance machine learning technology while also helping you manage the intricacies of this quickly developing industry.

Python, mainly its 3.x version, is very popular for machine learning. It lays a strong foundation. It has important libraries like NumPy, Pandas, Matplotlib, and Seaborn. They help a lot with data handling, studying, and making it visual. Coders can use popular spaces like Jupyter Notebooks, Google Colab, or others such as VSCode or PyCharm for smooth coding and trials. Main machine learning libraries are TensorFlow, PyTorch, and Scikit-learn. They help in building both complex deep learning models as well as usual machine learning methods.

For deep learning models, coders may use a strong GPU like NVIDIA's (like GTX 1060, RTX 2080). It helps to minimise long training times. The toolkit comes with many optional parts but offers a comprehensive guide. It aids in creating reliable and flexible machine learning systems. A successful machine learning system needs version control. It must consider hardware needs and keep its documentation up-to-date.

Hardware requirements:

Computer or server: A traditional computer with a multicore CPU (such an Intel Core i5 or higher) or a server with extensible processing capability.

Memory (RAM): You'll need at least 8 GB but if you're working with larger datasets or complex models, 16 GB is best.

Graphics Processing Unit (GPU): GPU Graphic Processing Unit: For deep learning, NVIDIA GPUs come highly recommended. A specialised GPU like NVIDIA Tesla, or a GTX 1060 and above from the RTX series, can bring down your training time.

Software requirements:

Operating System: Windows, macOS, Linux, or Ubuntu work great.

Python: Install Python 3.x using your OS package manager or from Python's official site. Use IDEs like PyCharm, VSCode, Jupyter Notebooks, or Google Colab for coding.

Libraries & Frameworks:

- The fundamental libraries for data manipulation and visualisation include NumPy, Pandas, Matplotlib, and Seaborn.
- For deep learning and machine learning applications, TensorFlow, PyTorch, and Scikit-learn are used. Use OpenCV for problems involving computer vision.

Libraries with Project Implementation Specialisations:

YOLOv4: To check bags using advanced video surveillance.

Power BI: For real-time data visualisation and analysis.

Web scraping: BeautifulSoup and Selenium are useful tools for web scraping tasks.

1 Airlines Operations Optimizations from passenger point of view

Passenger Satisfaction

```
#Importing basic libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
import sklearn
from sklearn.metrics import accuracy_score
from sklearn.neural_network import MLPClassifier
from sklearn.preprocessing import RobustScaler

!pip install catboost
!pip install CatBoostClassifieratboost

import pandas as pd
from sklearn.metrics import classification_report, roc_auc_score, accuracy_score, f1_score, precision_score, recall_score
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier, AdaBoostClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neural_network import MLPClassifier
from xgboost import XGBClassifier
from catboost import CatBoostClassifier
from tabulate import tabulate
```

Fig 1. Necessary libraries for Passenger satisfaction model

To execute the provided code, you will need a Python environment with the libraries pandas, numpy, matplotlib, warnings, sklearn, catboost, tabulate, and any additional libraries that the code requires installed. The code is designed to operate in a Jupyter Notebook environment and makes use of Google Colab for cloud-based execution. In order to install the required packages using the!pip install commands, ensure that your internet connection is stable. The algorithm expects to have access to this dataset so that machine learning models may be trained and evaluated.

```
classifiers = [  
    ("Logistic Regression", LogisticRegression()),  
    ("Decision Tree", DecisionTreeClassifier()),  
    ("Random Forest", RandomForestClassifier()),  
    ("Gradient Boosting", GradientBoostingClassifier()),  
    ("K-Nearest Neighbors", KNeighborsClassifier()),  
    ("Gaussian Naive Bayes", GaussianNB()),  
    ("Multi-layer Perceptron", MLPClassifier()),  
    ("XGBoost", XGBClassifier()),  
    ("CatBoost", CatBoostClassifier()),  
    ("AdaBoost", AdaBoostClassifier())  
]
```

Fig 2. Classifiers used for model training

Many classifiers, such as Gaussian Naive Bayes, K-Nearest Neighbours, Random Forest, Gradient Boosting, Decision Tree, AdaBoost, and XGBoost, are trained using the given code. Make sure you have sufficient processing power available because training different classifiers may demand a lot of it, especially when working with huge datasets.

A few classifiers from other machine learning packages, such as scikit-learn, are also used in the code. Make sure the Python environment has these libraries installed correctly. Lastly, for optimal code execution, ensure that your code is compatible with Google Colab or Jupyter Notebook environments.

Flight Fare Prediction

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
pd.set_option('display.max_columns',None)
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import Ridge
from sklearn import linear_model
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
import xgboost as xgb
from sklearn.neighbors import KNeighborsRegressor
from sklearn.ensemble import ExtraTreesRegressor
from sklearn.ensemble import BaggingRegressor
from sklearn.ensemble import GradientBoostingRegressor
```

Fig 3. Libraries Requirement for Flight Fare Prediction

You need a Python environment with the following libraries installed in order to execute the given code correctly: pandas, numpy, matplotlib, seaborn, warnings, scikit-learn, and xgboost. Make sure the Python environment has these libraries installed correctly. Additionally, the code preprocesses data using a few scikit-learn modules, including train_test_split, MinMaxScaler, and LabelEncoder. Verify if you can access these modules in your setup.

```
modelmlg = LinearRegression()
modeldcr = DecisionTreeRegressor()
modelbag = BaggingRegressor()
modelrfr = RandomForestRegressor()
modelsvr = SVR()
modelXGR = xgb.XGBRegressor()
modelKNN = KNeighborsRegressor(n_neighbors=5)
modelETR = ExtraTreesRegressor()
modelRE=Ridge()
modelLO=linear_model.Lasso(alpha=0.1)
modelGBR = GradientBoostingRegressor( learning_rate=0.1, n_estimators=100, subsample=1.0,criterion='friedman_mse')
```

Fig 4. Model Development Requirements

The code generates various regression models objects, including Gradient Boosting Regressor, KNeighborsRegressor, Ridge Regression, Lasso Regression, Extra Trees Regressor, Decision Tree Regressor, Bagging Regressor, RandomForest Regressor, Support

Vector Regressor (SVR), and Linear Regression. Verify that you have installed all of the libraries required for these models, and think about modifying the hyperparameters in accordance with the particular demands of the task.

Make sure that in Jupyter Notebook or any other Python environment of your choosing, the code runs as fast as feasible. The code is predicated on a fundamental knowledge of regression models and how they are used. perform sure you have access to the alternative dataset or perform the required code changes if one is chosen for the model's training.

2 Flight Delay Prediction

```
from fastai.tabular.all import *
df = pd.read_csv('../input/airlines-dataset-to-predict-a-delay/Airlines.csv')
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier, export_graphviz
from dtreeviz.trees import *
import graphviz
from sklearn.impute import SimpleImputer
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import LogisticRegression
```

Fig 5. Libraries required to implement flight delay prediction

Make sure you have configured the necessary Python environment before attempting to run the following code. Please make sure that fastai is installed because the code requires it to handle tabular data. Additionally, the pandas library is used for data processing, and scikit-learn is required for some machine learning techniques such as SVC, Logistic Regression, Random Forest Classifier, DecisionTree Classifier, and GradientBoostingClassifier. Please make sure your Python environment has these libraries installed correctly.

The code assumes that you have access to the CSV dataset located at `'../input/airlines-dataset-to-predict-a-delay/Airlines.csv'`. In the event that the dataset is saved someplace else, be careful to adjust the file path. Make sure you have all the required modules from fastai, dtreeviz, graphviz, and scikit-learn.

To ensure the best possible code execution, keep your Jupyter Notebook or any other Python environment compatible. Understanding machine learning principles and the scikit-learn package improves the interpretability and flexibility of the code to particular needs.

3 Real Time BI Dashboard

```
import requests
from bs4 import BeautifulSoup

response = requests.get(page_url)
content = BeautifulSoup(response.content, "html.parser")
content_reviews = content.find_all("article", class_ = lambda value: value and value.startswith("review-"))
review = content_reviews[0]
import pandas as pd

table = None
try_max = 2
try_count = 0
last_review_id = 0
while try_count < try_max:
    try:
        current_id = last_review_id + 1
        print(f"Will get review id {current_id}")
        url = f"{base_url}/page/{current_id}/?sortby=post_date%3AAsc&pagesize=1"
        review = get_review(url)
        review["id"] = current_id
        reviews_df = pd.DataFrame([review])
        try_count = 0 # Resets to get the next review
        last_review_id += 1 # Increments because will try to get next

        if table is not None:
            table = pd.concat([table, reviews_df])
        else:
            table = reviews_df
    except:
        try_count += 1
```

Fig 6. Data collection by Web Scraping

To successfully finish this Business Intelligence (BI) dashboard project, make sure your Python environment is set up with the required libraries. Because the code uses BeautifulSoup for HTML parsing and requests for sending HTTP requests, make sure these libraries are installed.¹

Information gathering from the SkyTrax website is the main objective of the web scraping capability. The code sample above is predicated on the web page having a correct HTML structure; if not, adjustments could be necessary.

¹ *Customer reviews - SKYTRAX*.

https://www.airlinequality.com/airline-reviews/british-airways/page/1/?sortby=post_date%3AAsc&pagesize=1.

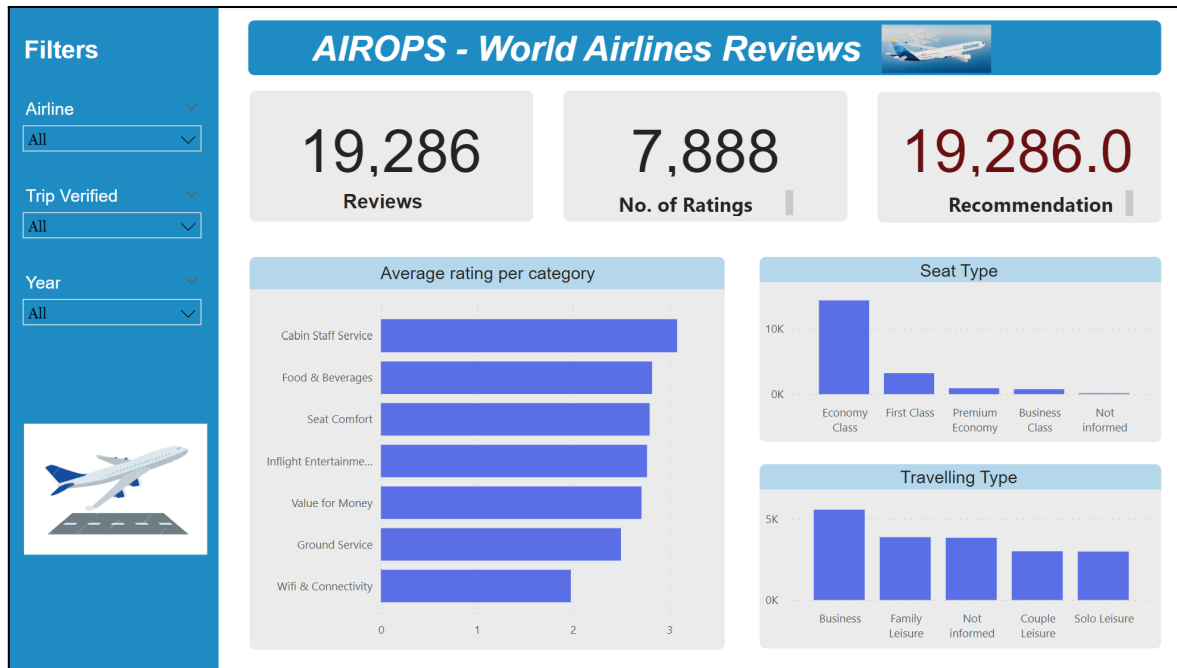


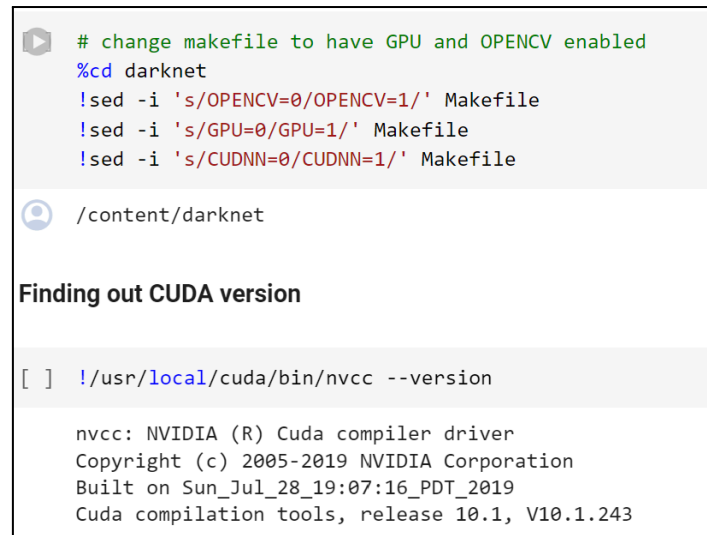
Fig 7. BI Dashboard Requirements

Make sure Power BI Desktop is installed on an appropriate and resource-rich Windows operating system before using it to create business intelligence dashboards². Make an account or sign in to Power BI, remembering that a Power BI Pro licence is needed for additional features. Make sure your internet connection is stable in order to retrieve data and use online services. It is anticipated that the data gathered by web scraping would be kept in a.csv file. To begin additional analysis and dashboard development, import the aggregated dataset into Power BI by following the instructions.

² Biswal, A. (2023) *What is Power BI?: Architecture, and Features Explained*. <https://www.simplilearn.com/tutorials/power-bi-tutorial/what-is-power-bi>.

4 Video Surveillance

YOLO4-based Advanced Baggage Scanning Algorithm:



```
# change makefile to have GPU and OPENCV enabled
%cd darknet
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile
!sed -i 's/GPU=0/GPU=1/' Makefile
!sed -i 's/CUDNN=0/CUDNN=1/' Makefile

/content/darknet

Finding out CUDA version

[ ] !/usr/local/cuda/bin/nvcc --version

nvcc: NVIDIA (R) Cuda compiler driver
Copyright (c) 2005-2019 NVIDIA Corporation
Built on Sun_Jul_28_19:07:16_PDT_2019
Cuda compilation tools, release 10.1, V10.1.243
```

Fig 8. Darnet & Cuda Version

Make sure your system has CUDA-version 10010, cuDNN 7.6.5, and at least one compatible GPU in order to utilise YOLO 4 for luggage scanning.³ Check that your computer has OpenCV 3.2.0 installed. The architecture of the YOLO model is made up of many convolutional layers in specific arrangements. It is necessary to meet the appropriate layer parameters, filter sizes, and input-output dimensions. When allocating GPU RAM for model optimisation, take into account variables such as time steps, batch size, and mini_batch. Make sure the weight file is compatible with the particular version of YOLO being used before downloading the YOLO 4 weights file.

³ Zvornicanin, E. and Zvornicanin, E. (2023) What is YOLO Algorithm? | Baeldung on Computer Science.
[https://www.baeldung.com/cs/yolo-algorithm#:~:text=3.-,You%20Only%20Look%20Once%20\(YOLO\),main%20reason%20for%20its%20popularity.](https://www.baeldung.com/cs/yolo-algorithm#:~:text=3.-,You%20Only%20Look%20Once%20(YOLO),main%20reason%20for%20its%20popularity.)

Weapon Detection:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import os

from xml.etree import ElementTree as et
import glob
boxes = []
labels = []

for xml in glob.glob('../input/firearm-object-detection/annotations/*.xml'):
```

Fig 9. Libraries Required for Weapon detection

Make sure you have installed the necessary libraries in your Python environment, such as numpy, pandas, matplotlib, albumentations, cv2, torch, and torchvision, before running the firearm object detection code. Arrange the XML files with object detection annotations for your dataset in the './input/firearm-object-detection/annotations/' directory. These XML files' item names and the corresponding bounding box coordinates (xmin, xmax, ymin, ymax) ought to follow a consistent format. Verify that the dataset also contains the pertinent picture files.

```
import torch
import cv2
import numpy as np
import os
import glob as glob

from xml.etree import ElementTree as et
from torch.utils.data import Dataset, DataLoader
import torch
import torch.nn as nn
import torch.nn.functional as F
from torch.utils.data import TensorDataset, DataLoader, Dataset, random_split
import torchvision.transforms as transforms
from torch.utils.data.sampler import SubsetRandomSampler
import torchvision
import torch.optim as optim
import torchvision.models as models
```

Fig 10. Model Development Requirements

Make sure the model weights are accessible or train the model before executing the code if the function makes use of a pre-trained model. Assuming you have installed the required versions of CUDA and cuDNN, using your suitable GPU might greatly speed up the training process.

Violence Detection:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
for dirname, _, filenames in os.walk('/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename), end = '\r')

import cv2
import random
import matplotlib.pyplot as plt
import json
from PIL import Image
from skimage.transform import resize
from collections import deque
import gc
```

Fig 11. Required libraries for Violence Detection

Verify that the following libraries are installed in your Python environment before running the aforementioned code: numpy, pandas, OpenCV (cv2), matplotlib, json, Pillow (PIL), scikit-image, TensorFlow, and scikit-learn. Setting aside more RAM is also a smart idea in case the code needs to load and manage a large number of photographs. Verify that the images are arranged correctly inside the dataset and that the code can access the dataset with the associated annotations and photos. If you intend to train the model, make sure your data is tagged and that the input shape is modified appropriately. Verify that the paths in the code used to load the data are accurate and correspond to the dataset's location.

```

from tensorflow.keras.layers import LSTM, Dense, Flatten, BatchNormalization, Dropout, InputLayer, MaxPooling2D, TimeDistributed, Conv2D
from tensorflow.keras.models import Sequential
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.callbacks import ModelCheckpoint

model = Sequential([
    InputLayer(shape),

    TimeDistributed(Conv2D(64, (3,3), activation = 'relu')),
    TimeDistributed(Dropout(0.2)),
    TimeDistributed(BatchNormalization()),
    TimeDistributed(MaxPooling2D((3,3))),

    TimeDistributed(Conv2D(32, (3,3), activation = 'relu')),
    TimeDistributed(Dropout(0.2)),
    TimeDistributed(BatchNormalization()),
    TimeDistributed(MaxPooling2D((3,3))),

    TimeDistributed(Conv2D(16, (3,3), activation = 'relu')),
    TimeDistributed(Dropout(0.2)),
    TimeDistributed(BatchNormalization()),
    TimeDistributed(MaxPooling2D((3,3))),

    TimeDistributed(Flatten()),
    LSTM(256, activation = 'relu', return_sequences=True),
    LSTM(128, activation = 'relu'),
    Dense(100, activation = 'relu'),
    Dense(1, activation = 'sigmoid')
])

model.summary()

```

Fig 12. Model Development of Violence Detection

Because the model architecture makes use of both convolutional and recurrent layers, it is faster to run the code on a GPU-equipped computer. Try shrinking the photos or changing the batch sizes if you are experiencing memory problems. You should be able to run the code for image classification using a convolutional and recurrent neural network combination after these prerequisites are satisfied.

People Counting and Tracking Using Neural Networks and OpenCV:

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import shutil
from sklearn.model_selection import train_test_split
import tensorflow as tf
from sklearn.model_selection import train_test_split
```

Fig 13. Libraries for People Counting

To run the provided code, you must first install the necessary libraries and set up a Python environment. Ascertain that matplotlib, shutil, scikit-learn, pandas, seaborn, numpy, and TensorFlow are available to you. To install any libraries that are missing, you can use a package management like pip. Ensure that your dataset is well-organized and readily available.

```

model = tf.keras.Sequential([

    tf.keras.layers.Conv2D(64, (3,3), input_shape=(480,640,3), activation=tf.keras.activations.relu),
    tf.keras.layers.MaxPool2D(2,2),
    tf.keras.layers.Conv2D(128, (3,3), activation=tf.keras.activations.relu),
    tf.keras.layers.MaxPool2D(2,2),
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation=tf.keras.activations.relu),
    tf.keras.layers.Dense(1)

])

model.compile(loss=tf.keras.losses.Huber(), optimizer=tf.keras.optimizers.Adam(), metrics=['mae'])
model.summary()

```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 478, 638, 64)	1792
max_pooling2d (MaxPooling2D)	(None, 239, 319, 64)	0
conv2d_1 (Conv2D)	(None, 237, 317, 128)	73856
max_pooling2d_1 (MaxPooling2D)	(None, 118, 158, 128)	0
dropout (Dropout)	(None, 118, 158, 128)	0
flatten (Flatten)	(None, 2386432)	0
dense (Dense)	(None, 128)	305463424
dense_1 (Dense)	(None, 1)	129

Total params: 305,539,201
 Trainable params: 305,539,201
 Non-trainable params: 0

Fig 14. Model Development for People Tracking & Counting

The scikit-learn `train_test_split` function is used in the code, so make sure your data is arranged appropriately. Check that the image data's dimensions correspond to the input shape specified in the Conv2D layer. TensorFlow must be installed in the correct version according to the model. If you have access to a GPU, you could consider using it.

After these issues are fixed, you may execute the code and utilise the image dataset to train a convolutional neural network. Make a note of the training summary that you receive from the model. To monitor the layer and architecture configurations, use the `summary()` function. Once the code has been executed in a Python environment suitable for deep learning assignments, adjust the network architecture and hyperparameters to suit your project's requirements.

References

Zvornicanin, E. and Zvornicanin, E. (2023) What is YOLO Algorithm? | Baeldung on Computer Science.

[https://www.baeldung.com/cs/yolo-algorithm#:~:text=3.-,You%20Only%20Look%20Once%20\(YOLO\),main%20reason%20for%20its%20popularity](https://www.baeldung.com/cs/yolo-algorithm#:~:text=3.-,You%20Only%20Look%20Once%20(YOLO),main%20reason%20for%20its%20popularity).

Customer reviews - SKYTRAX .

https://www.airlinequality.com/airline-reviews/british-airways/page/1/?sortBy=post_date%3AAsc&pagesize=1.

Biswal, A. (2023) What is Power BI?: Architecture, and Features Explained.

<https://www.simplilearn.com/tutorials/power-bi-tutorial/what-is-power-bi>.