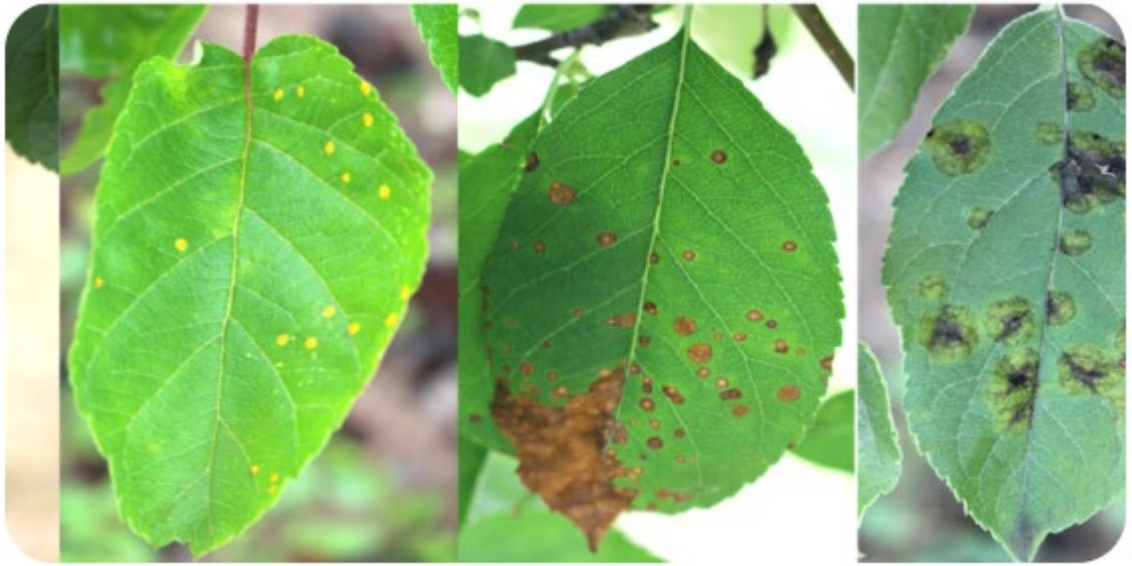
CNNの実習：キャグルの樹木の葉の病気の診断

https://www.kaggle.com/c/plant-pathology-2020-fgvc7

Plant Pathology 2020 - FGVC7

Identify the category of foliar diseases in apple trees



リンゴの木の葉のイメージで病気の種類を

判別するキャグルコンペティション

Plant Pathology 2020 - FGVC7

Identify the category of foliar diseases in apple trees

Overview Data Code Models Discussion Leaderboard Rules

Dataset Description

Given a photo of an apple leaf, can you accurately assess its health? This competition will challenge you to distinguish between leaves which are healthy, those which are infected with apple rust, those that have apple scab, and those with more than one disease.

Files
3645 files

Size
823.79 MB

Type
jpg, csv

License
Subject to Competition Rules

Files

train.csv

- `image_id`: the foreign key
- combinations: one of the target labels
- healthy: one of the target labels
- rust: one of the target labels
- scab: one of the target labels

リンゴの葉の写真が与えられたとき、あなたはその葉の健康状態を正確に判断できますか？このコンテストでは、健康な葉、さび病に感染している葉、リンゴかさぶたに感染している葉、複数の病気に感染している葉を見分けることに挑戦します。

データセットの概要

- Rust, scab, combination, healthyなど4つのタイプを判別
- 個々の画像サイズは (1365 × 2048)
- 学習画像の件数は1821件
- 性能評価指標はROC-AUC

Files

train.csv

- `image_id` : the foreign key
- combinations: one of the target labels
- healthy: one of the target labels
- rust: one of the target labels
- scab: one of the target labels

images

A folder containing the train and test images, in jpg format.

test.csv

- `image_id` : the foreign key

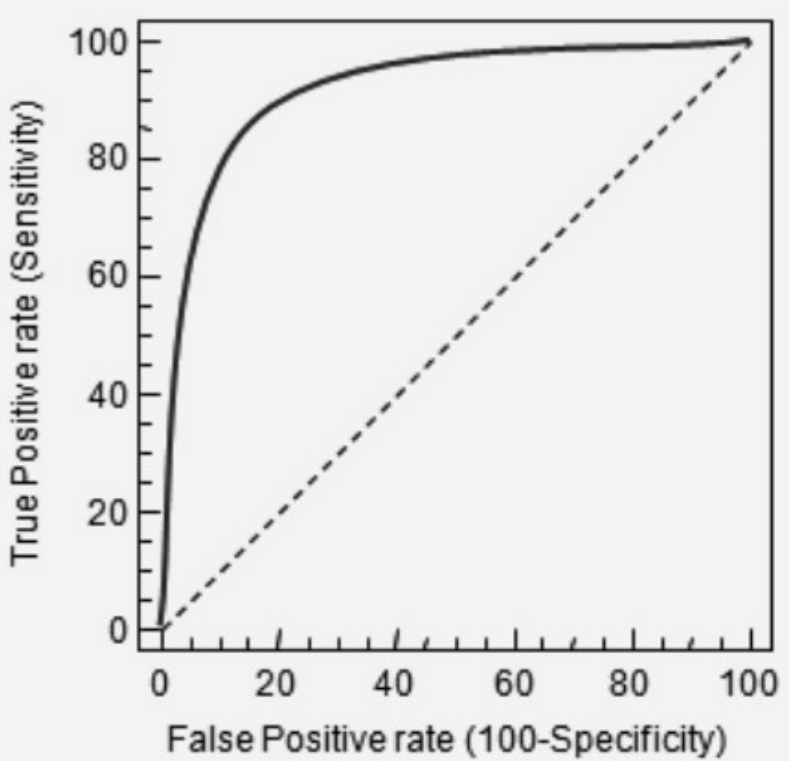
sample_submission.csv

- `image_id` : the foreign key
- combinations: one of the target labels
- healthy: one of the target labels
- rust: one of the target labels
- scab: one of the target labels

ROC-AUCとは

- ROC **曲線**(Receiver Operation Characteristic Curve)**とそれに基づいた**AUC(Area Under Curve)**スコアは、バイナリ分類の予測性能測定に使われる重要な指標である。**

- **一般的に医学分野で多く使用されますが、機械学習の分類モデルの予測性能を判断する指標として重要な評価指標でもあります。**



- ROC曲線は、**FPR(False Positive Rate)**が変化すると**TPR(True Positive Rate)**がどのように変化するかを示す曲線です。FPRをX軸に、TPRをY軸にすると、FPRの変化によるTPRの変化が曲線の形で現れます。

- 分類の性能指標として使用されるのは、ROC曲線の**面積**に基づいた**AUC値**で決定します。AUC値はROC曲線の下の面積を求めたもので、一般的に1に近いほど良い数値です。

ROC曲線

FPRの変化に伴うTPRの変化曲線

- TPRはTrue Positive Rateの略で、これは再現率(感度)を表します。

$$TPR = TP / (FN + TP)$$

- FPRは、実際のNegativeを誤ってPositiveに予測した割合を表します。

$$FPR = FP / (FP + TN)$$

- 閾値を1にするとFPRは0になります。
- 閾値を0にするとFPRは1である。

		予測した値（Predict class）	
		Negative(0)	Positive (1)
実際の値（Ground Truth）	Negative(0)	Negative Negative (True Negative) TN	Negative Positive (False Positive) FP
	Positive (1)	Positive Negative (False Negative) FN	Positive Positive (True Positive) TP

Evaluation

Submissions are evaluated on mean column-wise ROC AUC. In other words, the score is the average of the individual AUCs of each predicted column.

Submission File

For each image_id in the test set, you must predict a probability for each target variable. The file should contain a header and have the following format:

```
image_id,
test_0,0.25,0.25,0.25,0.25
test_1,0.25,0.25,0.25,0.25
test_2,0.25,0.25,0.25,0.25
etc.
```

提出する最終のフォーマット

kaggle

Search

Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More

FINE-GRAINED VISUAL CATEGORIZATION 7 · RESEARCH PREDICTION COMPETITION · 4 YEARS AGO

Plant Pathology 2020 - FGVC7

Identify the category of foliar diseases in apple trees

Overview Data Code Models Discussion Leaderboard Rules

Overview

Start
Mar 10, 2020

Close
May 27, 2020

Merger & Entry

kaggle

Search

Sign In

Register

Create

Home

Competitions

Datasets

Models

Code

Discussions

Learn

More

Code

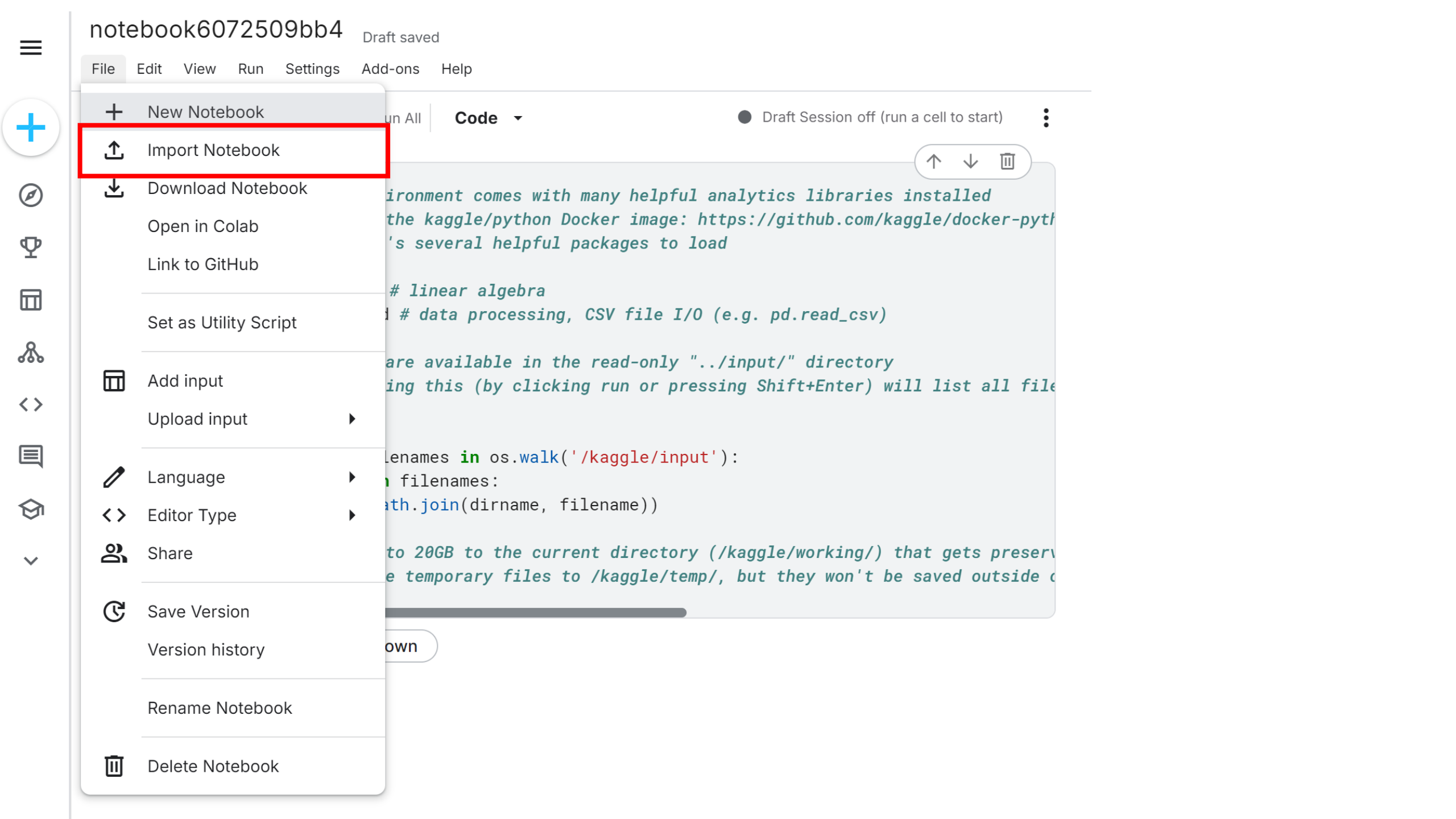
Explore and run machine learning code with Kaggle Notebooks.
Find help in the Documentation.

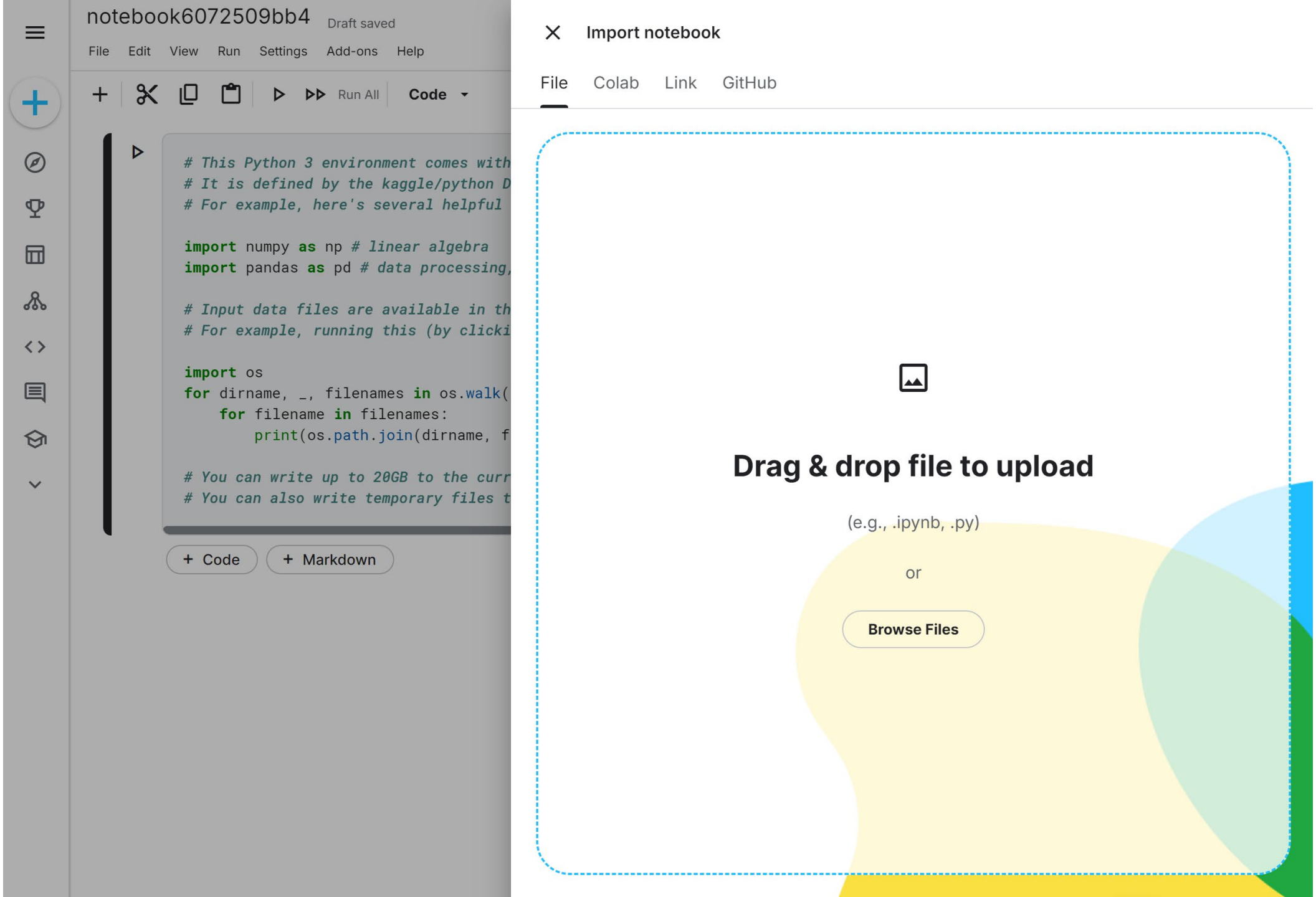
+ New Notebook

Search public notebooks

Filters

All notebooks | Recently Viewed | Python | R | Beginner | NLP | Random Forest | GPU | TPU | Competition notebook | Scheduled notebook


notebook6072509bb4
Draft saved
File
Edit
View
Run
Settings
Add-ons
Help
New Notebook
Import Notebook
Download Notebook
Open in Colab
Link to GitHub
Set as Utility Script
Add input
Upload input
Language
Editor Type
Share
Save Version
Version history
Rename Notebook
Delete Notebook
un All
Code
Draft Session off (run a cell to start)
ironment comes with many helpful analytics libraries installed
the kaggle/python Docker image: https://github.com/kaggle/docker-pyth
's several helpful packages to load
linear algebra
data processing, CSV file I/O (e.g. pd.read_csv)
are available in the read-only "../input/" directory
ing this (by clicking run or pressing Shift+Enter) will list all file
lenames in os.walk('/kaggle/input'):
filenames:
ath.join(dirname, filename))
to 20GB to the current directory (/kaggle/working/) that gets preserv
e temporary files to /kaggle/temp/, but they won't be saved outside
own


notebook6072509bb4
Draft saved
File Edit View Run Settings Add-ons Help
Run All
Code
# This Python 3 environment comes with
# It is defined by the kaggle/python D
# For example, here's several helpful
import numpy as np # linear algebra
import pandas as pd # data processing,
# Input data files are available in th
# For example, running this (by clicki
import os
for dirname, _, filenames in os.walk(
for filename in filenames:
print(os.path.join(dirname, f
You can write up to 20GB to the curr
You can also write temporary files t
+ Code
+ Markdown
Import notebook
File
Colab
Link
GitHub
Drag & drop file to upload
(e.g., .ipynb, .py)
or
Browse Files

Import notebook

File Colab Link GitHub

FILE

Plant_Pathology.ipynb

Import

plant_Pathology_01 Draft saved

File Edit View Run Settings Add-ons Help

Run All Markdown Draft Session off (run a cell to start)

ノートブックの名前を変更

+ Code + Markdown

[ ]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

[ ]:
```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

[ ]:
```
train_df.head()
```

+ Code + Markdown

```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

```
train_df.head()
```

Output

/kaggle/working

Session options

ACCELERATOR

None

None

GPU T4 x2

GPU P100

TPU VM v3-8

Pin to original environment

You won't get new packages, but your code is less likely to break. What is a notebook environment?

INTERNET

Internet on

plant_Pathology_01 Draft saved

File Edit View Run Settings Add-ons Help

Share Save Version 0

Run All Markdown Draft Session off (run a cell to start)

[ ]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

[ ]:
```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

[ ]:
```
train_df.head()
```

Notebook

Input

+ Add Input Upload

No input attached

Attach a Kaggle dataset, model, or competition

plant_Pathology_01 Draft saved

File Edit View Run Settings Add-ons Help

Share Save Version 0

Run All Markdown Draft Session off (run a cell to start)

[ ]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

+ Code + Markdown

[ ]:
```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

[ ]:
```
train_df.head()
```

Add Input

plant pathology

All

1,183 Results Relevance

Plant Pathology 2020 : EDA +...
Updated 4y ago
861 Upvotes

Plant Pathology 2020 - FGVC7
4y ago
1317 Teams · Research

Plant Pathology 2021 - FGVC8
3y ago
625 Teams · Research

Plant Pathology 2020 - Pytorch
Updated 4y ago

Please read the competition rules

By clicking on the "I Understand and Accept" button below, you agree to be bound by the competition rules for Plant Pathology 2020 - FGVC7.

any late, lost, damaged, misdirected, incomplete, illegible, undeliverable, or destroyed Submissions or entry materials due to system errors, failed, incomplete or garbled computer or other telecommunication transmission malfunctions, hardware or software failures of any kind, lost or unavailable network connections, typographical or system/human errors and failures, technical malfunction(s) of any telephone network or lines, cable connections, satellite transmissions, servers or providers, or computer equipment, traffic congestion on the Internet or at the Competition Website, or any combination thereof, which may limit a participant's ability to participate.

19. RIGHT TO CANCEL, MODIFY OR DISQUALIFY.

If for any reason the Competition is not capable of running as planned, including infection by computer virus, bugs, tampering, unauthorized intervention, fraud, technical failures, or any other causes which corrupt or affect the administration, security, fairness, integrity, or

I Understand and Accept

plant_Pathology_01 Draft saved

File Edit View Run Settings Add-ons Help

Share Save Version 0

Run All Markdown Draft Session off (run a cell to start)

[ ]:
```
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

[ ]:
```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

[ ]:
```
train_df.head()
```

Notebook

Input

ディレクトリとファイルの確認

COMPETITIONS

- Plant Pathology 2020 - FG
  - images
  - sample_submission.csv
  - test.csv
  - train.csv

Table of contents

plant-pathology-2020-fgvc7: Downloaded 705/1905 files: 157.72MiB/419.9MiB [37.5%]

```python
import numpy as np
import pandas as pd

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/plant-pathology-2020-fgvc7/sample_submission.csv
/kaggle/input/plant-pathology-2020-fgvc7/train.csv
/kaggle/input/plant-pathology-2020-fgvc7/test.csv
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1743.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_262.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1524.jpg
```

imagesのディレクトリに
TrainとTestファイルが
一緒に入っている

plant_Pathology_01 Draft saved

File Edit View Run Settings Add-ons Help

Share Save Version 0

Run All Code Draft Session (4m) HDD CPU RAM GPU GPU

```
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_4.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1055.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1438.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_296.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1364.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1074.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1260.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1759.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_399.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_871.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_630.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_573.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1142.jpg
```

COMPETITIONS

- Plant Pathology 2020 - FG
  - images
  - sample_submission.csv
  - test.csv
  - train.csv

train.csvファイルの確認

train.csv ×

train.csv (33.53 kB) /kaggle/input/plant-pathology-2020-fgvc7/train.csv Copy 5 of 5 columns

image_id	healthy	multiple_d...	rust	scab
Train_0	0	0	0	1
Train_1	0	1	0	0
Train_2	1	0	0	0

```
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_4.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1055.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1438.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_296.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1364.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1074.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1260.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1759.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_399.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_871.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_630.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Train_573.jpg
/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1142.jpg
```

COMPETITIONS

- Plant Pathology 2020 - FG
  - images
  - sample_submission.csv
  - test.csv
  - train.csv

test.csvファイルの確認

train.csv × test.csv ×

test.csv (17.11 kB) /kaggle/input/plant-pathology-2020-fgvc7/test.csv Copy 1 of 1 columns

image_id
Test_0
Test_1
Test_2

Prediction結果をここに入れる

[2]:

```
import numpy as np
import pandas as pd
import os

test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
train_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/train.csv")
```

testとtrainファイルをpandasで読み込む

+ Code + Markdown

```
train_df.head()
```

trainデータフレームを確認

[3]:

	image_id	healthy	multiple_diseases	rust	scab
0	Train_0	0	0	0	1
1	Train_1	0	1	0	0
2	Train_2	1	0	0	0
3	Train_3	0	0	1	0
4	Train_4	1	0	0	0

LabelがOne hot encodingされている

```
# healthy, multiple_diseases, rust, scab 列を合計しsumを作ってsumが1より大きいか, 0かを確認
train_df['sum'] = train_df['healthy'] + train_df['multiple_diseases'] + train_df['rust'] + train_df['scab']
train_df[(train_df['sum'] > 1) | (train_df['sum']==0)]
```

image_id	healthy	multiple_diseases	rust	scab	sum

One hot encodingされているLabelにエラーがあるか確認する

```
pd.set_option("max_colwidth", 100)

IMAGE_DIR = '/kaggle/input/plant-pathology-2020-fgvc7/images'
train_df['path'] = IMAGE_DIR + '/' + train_df['image_id'] + '.jpg'
train_df.head()
```

画像の絶対パスをDataFrameのpathに作成する

	image_id	healthy	multiple_diseases	rust	scab	sum	path
0	Train_0	0	0	0	1	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_0.jpg
1	Train_1	0	1	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1.jpg
2	Train_2	1	0	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_2.jpg
3	Train_3	0	0	1	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_3.jpg
4	Train_4	1	0	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_4.jpg

```python
def get_label(x):
    if x['healthy'] == 1:
        return 'healthy'
    elif x['multiple_diseases'] == 1:
        return 'multiple_diseases'
    elif x['rust'] == 1:
        return 'rust'
    elif x['scab'] == 1:
        return 'scab'
    else: return 'None'


train_df['label'] = train_df.apply(lambda x:get_label(x), axis=1)
train_df.head()
```

学習画像のlabelを分かりやすく作成する

:

	image_id	healthy	multiple_diseases	rust	scab	sum	path	label
0	Train_0	0	0	0	1	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_0.jpg	scab
1	Train_1	0	1	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_1.jpg	multiple_diseases
2	Train_2	1	0	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_2.jpg	healthy
3	Train_3	0	0	1	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_3.jpg	rust
4	Train_4	1	0	0	0	1	/kaggle/input/plant-pathology-2020-fgvc7/images/Train_4.jpg	healthy

```python
##labelごとに学習イメージが何枚か数える
print('train shape:', train_df.shape)
print('labelごとの学習画像数：')
train_df['label'].value_counts()
```

```
train shape: (1821, 8)
labelごとの学習画像数：
```

```
label
rust                 622
scab                 592
healthy              516
multiple_diseases     91
Name: count, dtype: int64
```

他のlabelに比べてデータ件数が少ない

元画像の可視化

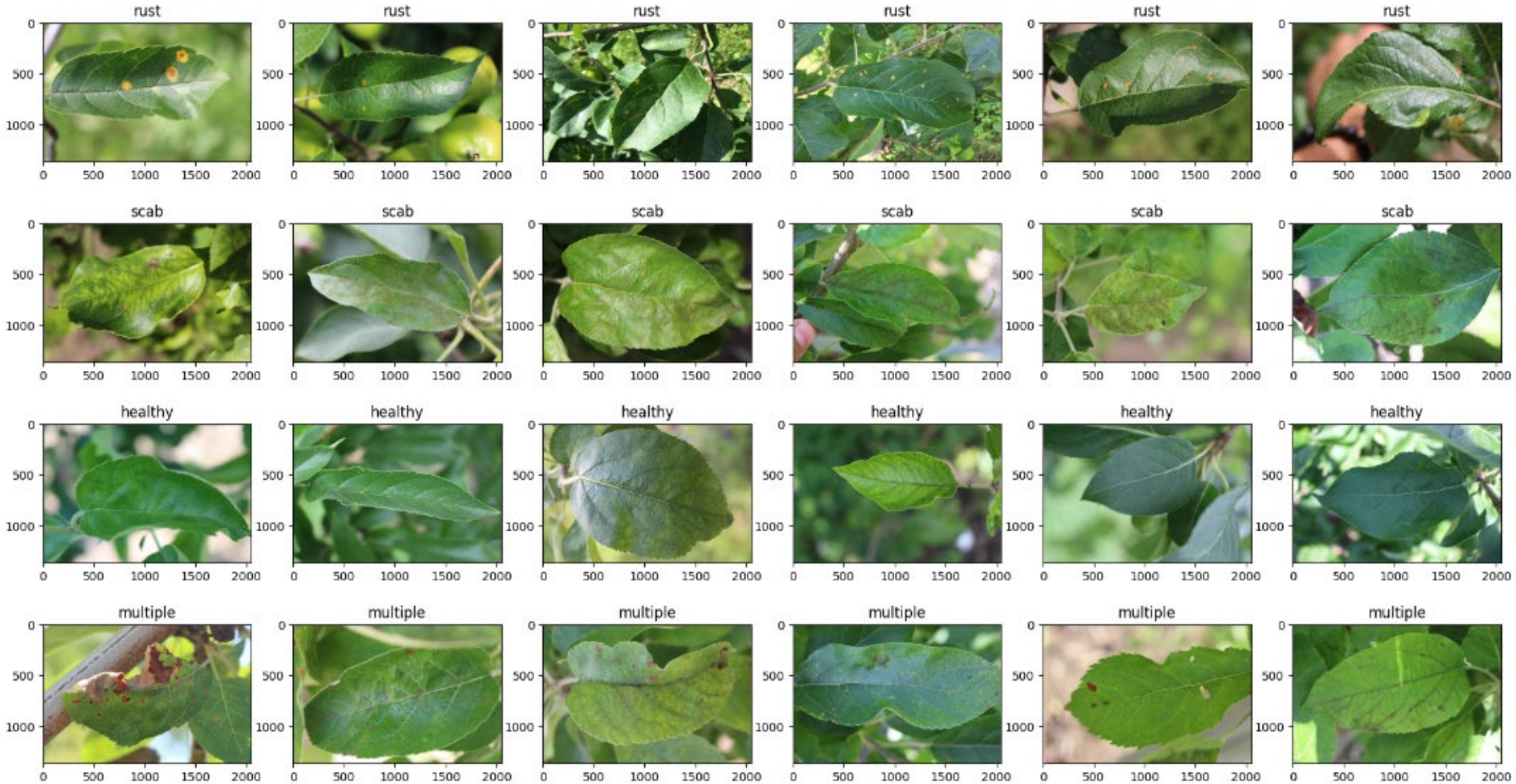
```
import seaborn as sns
import matplotlib.pyplot as plt
import cv2
%matplotlib inline

def show_grid_images(image_path_list, augmentor=None, ncols=4, title=None):
    figure, axs = plt.subplots(figsize=(22, 4), nrows=1, ncols=ncols)
    for i in range(ncols):
        image = cv2.cvtColor(cv2.imread(image_path_list[i]), cv2.COLOR_BGR2RGB)
        if augmentor is not None:
            image = augmentor(image=image)['image']
        axs[i].imshow(image)
        #axs[i].axis('off')
        axs[i].set_title(title)
        print(image.shape)

rust_image_list = train_df[train_df['label']=='rust']['path'].iloc[:6].tolist()
scab_image_list = train_df[train_df['label']=='scab']['path'].iloc[:6].tolist()
healthy_image_list = train_df[train_df['label']=='healthy']['path'].iloc[:6].tolist()
multiple_image_list = train_df[train_df['label']=='multiple_diseases']['path'].iloc[:6].tolist()

show_grid_images(rust_image_list, ncols=6, title='rust')
show_grid_images(scab_image_list, ncols=6, title='scab')
show_grid_images(healthy_image_list, ncols=6, title='healthy')
show_grid_images(multiple_image_list, ncols=6, title='multiple')
```

```
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
(1365, 2048, 3)
```


rust
rust
rust
rust
rust
rust
scab
scab
scab
scab
scab
scab
healthy
healthy
healthy
healthy
healthy
healthy
multiple
multiple
multiple
multiple
multiple
multiple

色が大事な情報になる

Augmentationを適用

*カットアウトのようなノイズは、葉の病原菌斑点と混同される可能性があるため、使用しない。
* 全体画像が緑系であり、病原菌斑点が特定の色（オレンジ，ブラウンなど）を持っているため、色の変化は適用しない。
* 全体的に判別したい葉が画像全体の中央に位置しているので、スケール等の適用を検討。

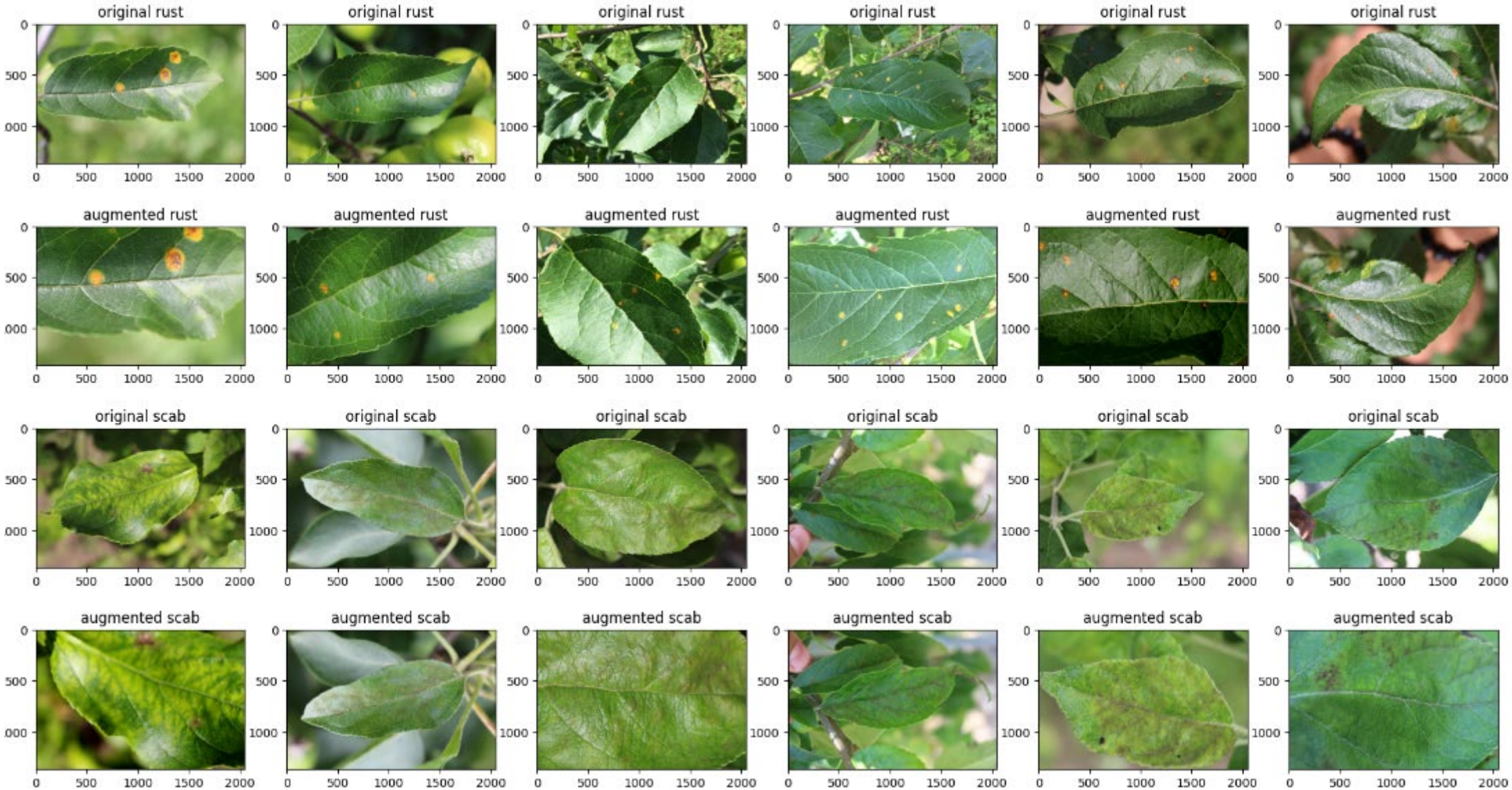
```
import albumentations as A

augmentor_01 = A.Compose([
    A.HorizontalFlip(p=0.5),
    A.VerticalFlip(p=0.5),
    A.ShiftScaleRotate(scale_limit=(0.7, 0.9), p=0.5, rotate_limit=30),
    A.RandomBrightnessContrast(brightness_limit=(-0.2, 0.2), contrast_limit=(-0.2, 0.2), p=0.5),
    A.Blur(p=0.2)
])

show_grid_images(rust_image_list, augmentor=None, ncols=6, title='original rust')
show_grid_images(rust_image_list, augmentor=augmentor_01, ncols=6, title='augmented rust')

show_grid_images(scab_image_list, augmentor=None, ncols=6, title='original scab')
show_grid_images(scab_image_list, augmentor=augmentor_01, ncols=6, title='augmented scab')
```

自分が自由に工夫して適用する


original rust
original rust
original rust
original rust
original rust
original rust
augmented rust
augmented rust
augmented rust
augmented rust
augmented rust
augmented rust
original scab
original scab
original scab
original scab
original scab
original scab
augmented scab
augmented scab
augmented scab
augmented scab
augmented scab
augmented scab

SequenceベースのDataset生成

- 従来はimage sizeが高さと幅が同じでしたが、今回は高さと幅が違う場合があることを考慮してimage_sizeをtupleで入力する。

- opencvのresize()は引数で画像サイズを入力しますが、横x縦(幅x高さ)の概念で入力します。画像配列の場合は行x列(高さx幅)なので、resize()呼び出し時にこれを考慮する。

* キャグルコンペティションにテストデータの結果をsubmitするため、テスト日付のLabelがない。 そのため、Datasetのlabel_batchの値がNoneになる可能性があるため、コードの修正が必要。

```python
from sklearn.model_selection import train_test_split
from tensorflow.keras.utils import Sequence
import sklearn
import cv2

#入力引数image_filenames、labelsは、全てnumpy arrayで入る。image sizeは(高さ、幅)で修正。
class Plant_Dataset(Sequence):
    def __init__(self, image_filenames, labels, image_size=(224, 224), batch_size=64,
                 augmentor=None, shuffle=False, pre_func=None):
        '''
        パラメータの説明
        image_filenames: opencvでimageをロードするファイルの絶対パス。
        labels: 該当するimageのラベル
        batch_size : __getitem__(self, index)呼び出しごとに取り込むデータのバッチ数。
        augmentor: albumentationsのオブジェクト。
        shuffle: 学習データの場合、epoch終了時ごとにデータを混ぜるかどうかを決める。
        '''
        # オブジェクト生成引数として入ってきた値をオブジェクト内部変数に割り当てる。
        self.image_filenames = image_filenames
        self.labels = labels
        self.image_size = image_size
        self.batch_size = batch_size
        self.augmentor = augmentor
        self.pre_func = pre_func
        # train dataの場合
        self.shuffle = shuffle
        if self.shuffle:
            # オブジェクト生成する時一回Shuffleする
            #self.on_epoch_end()
            pass
```

tuple

```python
# Sequenceを継承するDatasetは batch_size 単位で入力データを処理する
# __len__()は全体データ数が与えられた時、batch_size単位でデータを何回返すかを表す
def __len__(self):
    return int(np.ceil(len(self.image_filenames) / self.batch_size))

def __getitem__(self, index):
    # indexは何回目のbatchかを表す.
    # batch_sizeごとに順次的にデータを持ってくるように記述する。
    image_name_batch = self.image_filenames[index*self.batch_size:(index+1)*self.batch_size]
    if self.labels is not None:
        label_batch = self.labels[index*self.batch_size:(index+1)*self.batch_size]

    # label_batchが Noneになる場合
    else:
        label_batch = None
    # もし、オブジェクト生成引数にalbumentationで作ったaugmentorが与えられた場合、下記のようにaugmentorを利用してimage変換する。
    # albumentationsは個々のimageしか変換できないので、batch_sizeだけ割り当てられたimage_name_batchを1件ずつiterationしながら変換
    # image_batch 配列は float32 に設定。
    image_batch = np.zeros((image_name_batch.shape[0], self.image_size[0], self.image_size[1], 3), dtype='float32')


    for image_index in range(image_name_batch.shape[0]):
        image = cv2.cvtColor(cv2.imread(image_name_batch[image_index]), cv2.COLOR_BGR2RGB)
        if self.augmentor is not None:
            image = self.augmentor(image=image)['image']
        #resize適用. opencvのresizeは (幅, 縦)の概念. 配列は (縦, 幅)になるので注意しながらopencvのresizeに引数を入力する.
        image = cv2.resize(image, (self.image_size[1], self.image_size[0]))
        # もし preprocessing_inputが pre_func引数に入るとこれを利用し scalingを適用.
        if self.pre_func is not None:
            image = self.pre_func(image)

        image_batch[image_index] = image

    if self.labels is None:
        return image_batch
    else:
        return image_batch, label_batch
```

学習データ用DataFrameから学習用/検証用画像の絶対パスとLabelを抽出し、これをDatasetとして作成。

- すでに学習用DataFrameに'healthy'、'multiple_diseases'、'rust'、'scab'の順にワンホットエンコーディング(One hot encoding)されています。
- キャグルでテストデータ予測した結果を'healthy','multiple_diseases','rust','scab'の順番で提出を要求するので、これを別途に再度ワンホットエンコーディングしてはいけません。
- Augmentation は先に生成した augmentor_01 を適用。pre_funcはxception用のPreprocessing関数を適用。

+ Code + Markdown

```
sample_df = pd.read_csv('/kaggle/input/plant-pathology-2020-fgvc7/sample_submission.csv')
sample_df.head()
```

]:

	image_id	healthy	multiple_diseases	rust	scab
0	Test_0	0.25	0.25	0.25	0.25
1	Test_1	0.25	0.25	0.25	0.25
2	Test_2	0.25	0.25	0.25	0.25
3	Test_3	0.25	0.25	0.25	0.25
4	Test_4	0.25	0.25	0.25	0.25

```python
from sklearn.model_selection import train_test_split

def get_train_valid(train_df, valid_size=0.2, random_state=2021):
    train_path = train_df['path'].values
    train_label = train_df[['healthy', 'multiple_diseases', 'rust', 'scab']].values

    tr_path, val_path, tr_label, val_label = train_test_split(train_path, train_label, test_size=valid_size, random_state=random_state)
    print('tr_path shape:', tr_path.shape, 'tr_label shape:', tr_label.shape, 'val_path shape:', val_path.shape, 'val_label shape:', val_label.shape)
    return tr_path, val_path, tr_label, val_label
```

+ Code + Markdown

```python
from tensorflow.keras.applications.xception import preprocess_input as xcp_preprocess_input
from tensorflow.keras.applications.efficientnet import preprocess_input as eff_preprocess_input

# image size는 224x224로 Dataset 생성.
IMAGE_SIZE = (224, 224)
BATCH_SIZE = 64

tr_path, val_path, tr_label, val_label = get_train_valid(train_df, valid_size=0.2, random_state=2021)

tr_ds = Plant_Dataset(tr_path, tr_label, image_size=IMAGE_SIZE, batch_size=BATCH_SIZE,
                      augmentor=augmentor_01, shuffle=True, pre_func=xcp_preprocess_input)
val_ds = Plant_Dataset(val_path, val_label, image_size=IMAGE_SIZE, batch_size=BATCH_SIZE,
                       augmentor=None, shuffle=False, pre_func=xcp_preprocess_input)

tr_image_batch, tr_label_batch = next(iter(tr_ds))
val_image_batch, val_label_batch = next(iter(val_ds))
print(tr_image_batch.shape, val_image_batch.shape, tr_label_batch.shape, val_label_batch.shape)
print(tr_image_batch[0], val_image_batch[0])
```

create_model() 関数を生成
* resnet50v2, xception, efficientnetb0~b7 などの Pretrained モデルを生成

```
from tensorflow.keras.models import Sequential , Model
from tensorflow.keras.layers import Input, Dense , Conv2D , Dropout , Flatten , Activation, MaxPooling2D , GlobalAveragePooling2D
from tensorflow.keras.optimizers import Adam , RMSprop
from tensorflow.keras.regularizers import l2
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.callbacks import ReduceLROnPlateau , EarlyStopping , ModelCheckpoint , LearningRateScheduler
from tensorflow.keras.metrics import AUC

from tensorflow.keras.applications import Xception, ResNet50V2, EfficientNetB0, EfficientNetB1, EfficientNetB2, EfficientNetB3
from tensorflow.keras.applications import EfficientNetB4, EfficientNetB5, EfficientNetB6, EfficientNetB7
import tensorflow as tf


def create_model(model_type='efficientnetb0', in_shape=(224, 224, 3), n_classes=4):
    input_tensor = Input(shape=in_shape)

    if model_type == 'resnet50v2':
        base_model = tf.keras.applications.ResNet50V2(include_top=False, weights='imagenet', input_tensor=input_tensor)
    elif model_type == 'xception':
        base_model = tf.keras.applications.Xception(include_top=False, weights='imagenet', input_tensor=input_tensor)
    elif model_type == 'efficientnetb0':
        base_model = tf.keras.applications.EfficientNetB0(include_top=False, weights='imagenet', input_tensor=input_tensor)
    elif model_type == 'efficientnetb1':
        base_model = tf.keras.applications.EfficientNetB1(include_top=False, weights='imagenet', input_tensor=input_tensor)
    elif model_type == 'efficientnetb2':
        base_model = tf.keras.applications.EfficientNetB2(include_top=False, weights='imagenet', input_tensor=input_tensor)
```

```
elif model_type == 'efficientnetb3':
    base_model = tf.keras.applications.EfficientNetB3(include_top=False, weights='imagenet', input_tensor=input_tensor)
elif model_type == 'efficientnetb4':
    base_model = tf.keras.applications.EfficientNetB4(include_top=False, weights='imagenet', input_tensor=input_tensor)
elif model_type == 'efficientnetb5':
    base_model = tf.keras.applications.EfficientNetB5(include_top=False, weights='imagenet', input_tensor=input_tensor)
elif model_type == 'efficientnetb6':
    base_model = tf.keras.applications.EfficientNetB6(include_top=False, weights='imagenet', input_tensor=input_tensor)
elif model_type == 'efficientnetb7':
    base_model = tf.keras.applications.EfficientNetB7(include_top=False, weights='imagenet', input_tensor=input_tensor)
```

```
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(1024, activation='relu')(x)
x = Dropout(0.5)(x)
preds = Dense(units=n_classes, activation='softmax')(x)
model = Model(inputs=input_tensor, outputs=preds)


return model
```

xception　モデルを生成して学習を行う。

- image sizeは224x224で生成。
- Learning Rate SchedulerはReduceLROnPlateauで、初期Learning Rateは0.0001に設定。
- epochsは10回のみ設定。
- metricsはROC-AUCを設定

+ Code　+ Markdown

```
from tensorflow.keras.metrics import AUC

xcp_model_01 = create_model(model_type='xception', in_shape=(224, 224, 3))
xcp_model_01.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=[AUC()])

#  3回のiterationでvalidation lossが改善されない場合、learning rateを従来のlearning rate * 0.2に減らす。
rlr_cb = ReduceLROnPlateau(monitor='val_loss', factor=0.2, patience=3, mode='min', verbose=1)
#  10回のiterationでvalidation lossが改善されない場合は、これ以上学習せずに終了します。
ely_cb = EarlyStopping(monitor='val_loss', patience=10, mode='min', verbose=1)

history = xcp_model_01.fit(tr_ds, epochs=10, steps_per_epoch=int(np.ceil(tr_path.shape[0]/BATCH_SIZE)),
                validation_data=val_ds, validation_steps=int(np.ceil(val_path.shape[0]/BATCH_SIZE)),
                callbacks=([rlr_cb, ely_cb]), verbose=1)
```

```
Epoch 1/10
```

```
Epoch 2/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9517 - loss: 0.5511 - val_auc_1: 0.9217 - val_loss: 0.8791 - learning_rate: 1.0000e-04
Epoch 3/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 47s 2s/step - auc_1: 0.9722 - loss: 0.3987 - val_auc_1: 0.9471 - val_loss: 0.6718 - learning_rate: 1.0000e-04
Epoch 4/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9891 - loss: 0.2457 - val_auc_1: 0.9739 - val_loss: 0.3922 - learning_rate: 1.0000e-04
Epoch 5/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 47s 2s/step - auc_1: 0.9943 - loss: 0.1843 - val_auc_1: 0.9744 - val_loss: 0.3551 - learning_rate: 1.0000e-04
Epoch 6/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9950 - loss: 0.1630 - val_auc_1: 0.9683 - val_loss: 0.4173 - learning_rate: 1.0000e-04
Epoch 7/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 47s 2s/step - auc_1: 0.9952 - loss: 0.1520 - val_auc_1: 0.9777 - val_loss: 0.3000 - learning_rate: 1.0000e-04
Epoch 8/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9980 - loss: 0.1102 - val_auc_1: 0.9803 - val_loss: 0.3092 - learning_rate: 1.0000e-04
Epoch 9/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9983 - loss: 0.0978 - val_auc_1: 0.9806 - val_loss: 0.3230 - learning_rate: 1.0000e-04
Epoch 10/10
23/23 ━━━━━━━━━━━━━━━━━━━━ 0s 1s/step - auc_1: 0.9987 - loss: 0.0804
Epoch 10: ReduceLROnPlateau reducing learning rate to 1.9999999494757503e-05.
23/23 ━━━━━━━━━━━━━━━━━━━━ 46s 2s/step - auc_1: 0.9987 - loss: 0.0804 - val_auc_1: 0.9757 - val_loss: 0.3999 - learning_rate: 1.0000e-04
```

テストデータでPlantの病気を予測してキャグルに提出するsubmit csvファイルを作成します。

- テスト用DataFrameに画像パスを追加。
- テスト用Datasetを生成します。labelはテストデータでは分からないのでNoneで入力。

```
sample_df = pd.read_csv('/kaggle/input/plant-pathology-2020-fgvc7/sample_submission.csv')
sample_df.head()
```

	image_id	healthy	multiple_diseases	rust	scab
0	Test_0	0.25	0.25	0.25	0.25
1	Test_1	0.25	0.25	0.25	0.25
2	Test_2	0.25	0.25	0.25	0.25
3	Test_3	0.25	0.25	0.25	0.25
4	Test_4	0.25	0.25	0.25	0.25

```
IMAGE_DIR = '/kaggle/input/plant-pathology-2020-fgvc7/images'
test_df = pd.read_csv("../input/plant-pathology-2020-fgvc7/test.csv")
test_df['path'] = IMAGE_DIR + '/' + test_df['image_id'] + '.jpg'

test_df.head(10)
```

絶対パースを追加

	image_id	path
0	Test_0	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_0.jpg
1	Test_1	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_1.jpg
2	Test_2	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_2.jpg
3	Test_3	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_3.jpg
4	Test_4	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_4.jpg
5	Test_5	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_5.jpg
6	Test_6	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_6.jpg
7	Test_7	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_7.jpg
8	Test_8	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_8.jpg
9	Test_9	/kaggle/input/plant-pathology-2020-fgvc7/images/Test_9.jpg

```
# テスト用Datasetを生成し、これを利用してmodelのpredict()を呼び出して画像予測を実行します。

test_path = test_df['path'].values
# labelsは Noneを入力して Datasetを生成.
test_ds = Plant_Dataset(image_filenames=test_path, labels=None, image_size=IMAGE_SIZE, batch_size=BATCH_SIZE,
                        augmentor=None, shuffle=False, pre_func=xcp_preprocess_input)
#predict()から 予測する作業を行う.
preds = xcp_model_01.predict(test_ds)
```

テストはlabelがないのでNoneに設定

+ Code + Markdown

```
preds_df = pd.DataFrame(preds)
preds_df.columns = ['healthy', 'multiple_diseases', 'rust', 'scab']
preds_df.head()
```

	healthy	multiple_diseases	rust	scab
0	1.052231e-05	0.000134	9.998443e-01	1.091581e-05
1	4.836722e-07	0.000093	9.999059e-01	2.637625e-07
2	1.032351e-07	0.000002	7.365213e-08	9.999974e-01
3	9.992207e-01	0.000016	7.503788e-04	1.260714e-05
4	1.137724e-05	0.000193	9.997917e-01	4.101276e-06

```
# 予測した結果に基づき、別の結果DataFrameを作成。
preds_df = pd.DataFrame(preds)
preds_df.columns = ['healthy', 'multiple_diseases', 'rust', 'scab']
# テスト用DataFrameに上記で生成した結果DataFrameを合成し、これを利用してsubmit用DataFrameを生成します。
submit_df = pd.concat([test_df['image_id'], preds_df], axis = 1)
submit_df.head()
```

:

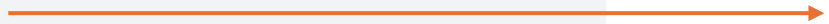
	image_id	healthy	multiple_diseases	rust	scab
0	Test_0	1.052231e-05	0.000134	9.998443e-01	1.091581e-05
1	Test_1	4.836722e-07	0.000093	9.999059e-01	2.637625e-07
2	Test_2	1.032351e-07	0.000002	7.365213e-08	9.999974e-01
3	Test_3	9.992207e-01	0.000016	7.503788e-04	1.260714e-05
4	Test_4	1.137724e-05	0.000193	9.997917e-01	4.101276e-06

提出する最終フォーマットをcsvファイルに作成

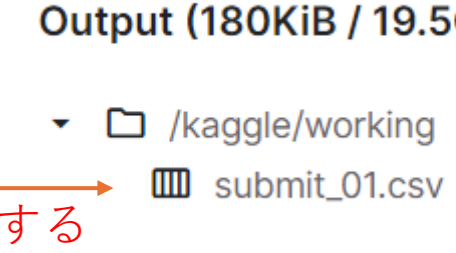
キャグル提出用CSVを作成した後、キャグルに提出し、テスト性能を確認する

+ Code + Markdown

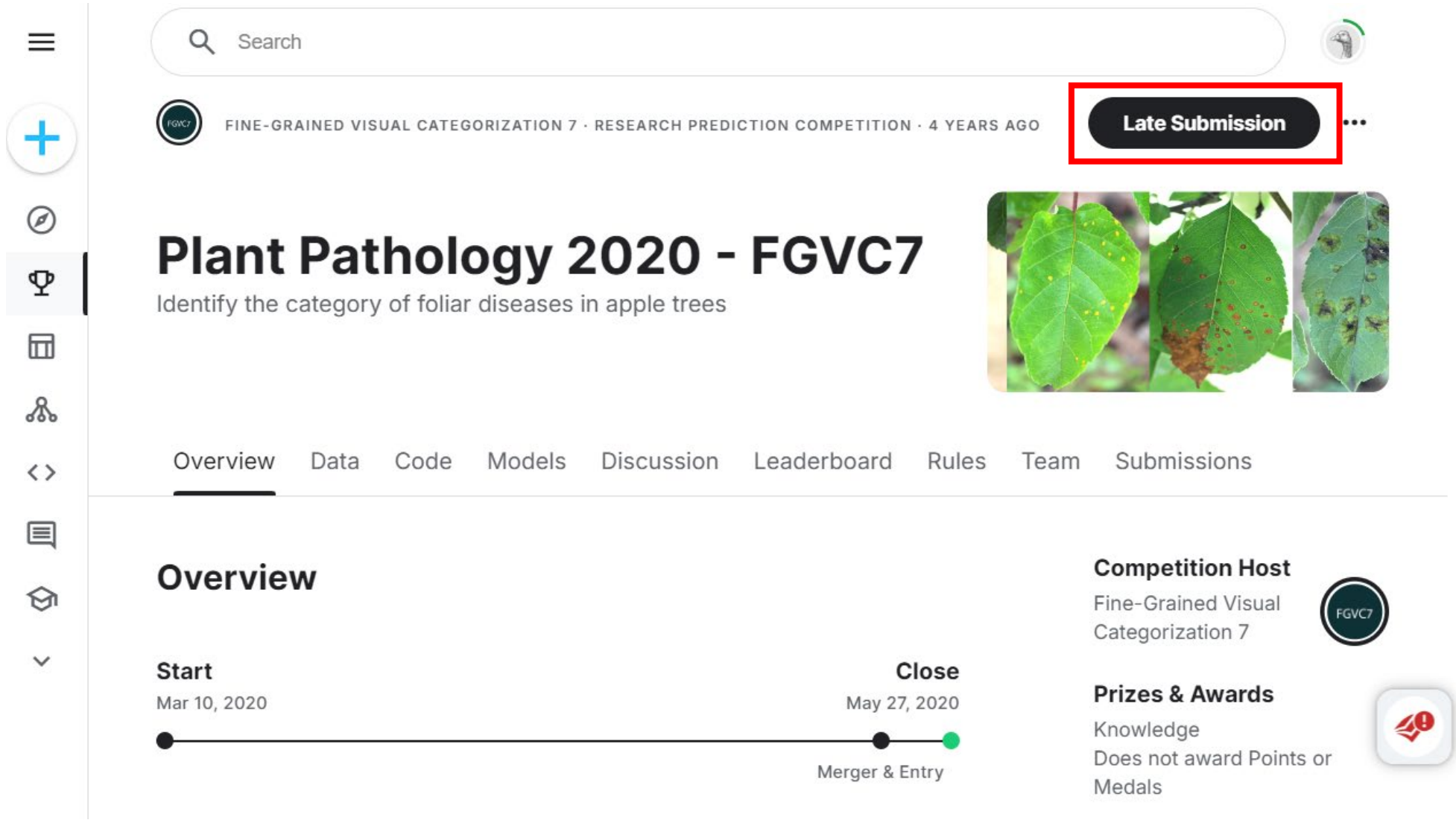
```
submit_df.to_csv('submit_01.csv', index=False)
```



ダウンロードする



Kaggleへ提出


Search
FINE-GRAINED VISUAL CATEGORIZATION 7 · RESEARCH PREDICTION COMPETITION · 4 YEARS AGO
Late Submission
Plant Pathology 2020 - FGVC7
Identify the category of foliar diseases in apple trees
Overview
Data
Code
Models
Discussion
Leaderboard
Rules
Team
Submissions
Overview
Start
Mar 10, 2020
Close
May 27, 2020
Merger & Entry
Competition Host
Fine-Grained Visual Categorization 7
Prizes & Awards
Knowledge
Does not award Points or Medals

Submit to Competition

File Upload | Notebook

Plant Pathology 2020 - FGVC7

You have 100 submissions remaining today. This resets in 8 hours.

Drag and drop file to upload

(e.g., .csv, .zip, .gz, .7z)

Uploaded File

submit_01.csv (105 KiB)

Cancel Submit

性能向上

Submission and Description	Private Score	Public Score	Selected
submit_01.csv Complete (after deadline) · now · First submission	0.91530	0.91184	☐

#	△	Team	Members	Score	Entries	Last	Solution
1	▲ 17	Alipay Tian Suan Security Lab		0.98445	115	4y	
2	▲ 94	Shenlan		0.98182	15	4y	
3	▲ 103	Clay		0.98089	167	4y	
4	▲ 61	Mack Tang		0.98064	28	4y	
5	▲ 9	Profit!		0.98060	361	4y	