Pokemon_Final_Project

August 14, 2022

[4]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call
drive.mount("/content/drive", force_remount=True).
```

[5]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
import matplotlib.image as mpimg
import zipfile
import os
from pathlib import Path
import re
import tensorflow as tf
from tensorflow import keras
from keras import layers
from PIL import Image
import plotly.express as px
import glob
```

[6]:
```
data = pd.read_csv('/content/pokemon.csv')
```

[8]:
```
directory = r'/content/drive/MyDrive/images/images/'
for filename in os.listdir(directory):
    if filename.endswith('.jpg'):
        prefix = filename.split('.jpg')[0]
        os.rename(os.path.join(directory, filename), os.path.join(directory,␣
 ↪prefix+'.png'))
    else:
        continue
```

[9]:
```
train_dir = r'/content/drive/MyDrive/images/images/'
train_path = Path(train_dir)
```

[10]:
```
files = list(train_path.glob('*.png'))
name = [os.path.split(x)[1] for x in list(train_path.glob('*.png'))]
```

```
pokemon_image_df = pd.concat([pd.Series(name, name='Name'), pd.Series(files,␣
↪name='Filepath').astype(str)], axis=1)
pokemon_image_df['Name'] = pokemon_image_df['Name'].apply(lambda x: re.sub(r'\.
↪\w+$', '', x))
pokemon_image_df
```

[10]:

	Name	Filepath
0	aurorus	/content/drive/MyDrive/images/images/aurorus.png
1	arcanine	/content/drive/MyDrive/images/images/arcanine.png
2	abra	/content/drive/MyDrive/images/images/abra.png
3	barbaracle	/content/drive/MyDrive/images/images/barbaracl…
4	amoonguss	/content/drive/MyDrive/images/images/amoonguss…
..	…	…
804	vikavolt	/content/drive/MyDrive/images/images/vikavolt.png
805	wishiwashi-solo	/content/drive/MyDrive/images/images/wishiwash…
806	yungoos	/content/drive/MyDrive/images/images/yungoos.png
807	type-null	/content/drive/MyDrive/images/images/type-null…
808	zeraora	/content/drive/MyDrive/images/images/zeraora.png

[809 rows x 2 columns]

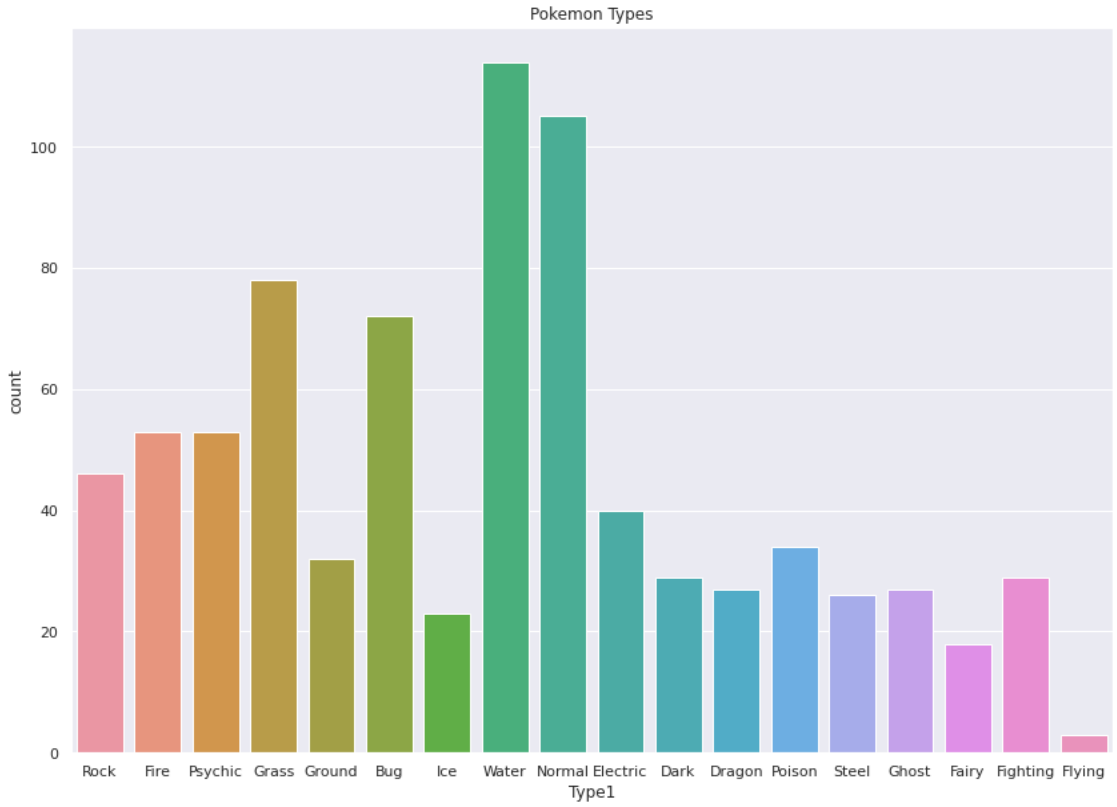
[11]:
```
#list(train_path.glob('*.png'))
comb_df = pokemon_image_df.merge(data, on='Name')
comb_df = comb_df.drop(['Name','Type2'], axis=1)
comb_df
```

[11]:

	Filepath	Type1
0	/content/drive/MyDrive/images/images/aurorus.png	Rock
1	/content/drive/MyDrive/images/images/arcanine.png	Fire
2	/content/drive/MyDrive/images/images/abra.png	Psychic
3	/content/drive/MyDrive/images/images/barbaracl…	Rock
4	/content/drive/MyDrive/images/images/amoonguss…	Grass
..	…	…
804	/content/drive/MyDrive/images/images/vikavolt.png	Bug
805	/content/drive/MyDrive/images/images/wishiwash…	Water
806	/content/drive/MyDrive/images/images/yungoos.png	Normal
807	/content/drive/MyDrive/images/images/type-null…	Normal
808	/content/drive/MyDrive/images/images/zeraora.png	Electric

[809 rows x 2 columns]

[13]:
```
sns.set(style="darkgrid")
sns.countplot(x="Type1", data=comb_df).set(title='Pokemon Types')
fig = plt.gcf()
# Change seaborn plot size
fig.set_size_inches(14, 10)
```



[14]:
```
path = r'/content/drive/MyDrive/images/images/'
fig,(ax1, ax2, ax3, ax4, ax5, ax6, ax7, ax8) = plt.subplots(1, 8, figsize=(15,␣
 ↪10))
ax = [ax1, ax2, ax3, ax4, ax5, ax6, ax7, ax8]
for i in range(8):
    img = mpimg.imread(path+data['Name'][i**3]+'.png', 0)
    ax[i].imshow(img)
    ax[i].set_title(data['Name'][i**3])
    ax[i].axis('off')
plt.tight_layout()
plt.show()
```

[19]:
```
firewater = comb_df.query("Type1 == 'Fire' | Type1 == 'Water'")
firewater

print("Number of Fire Types:", len(firewater[firewater['Type1'] == 'Fire']))
print("Number of Water Types:", len(firewater[firewater['Type1'] == 'Water']))
```

```
Number of Fire Types: 53
Number of Water Types: 114
```

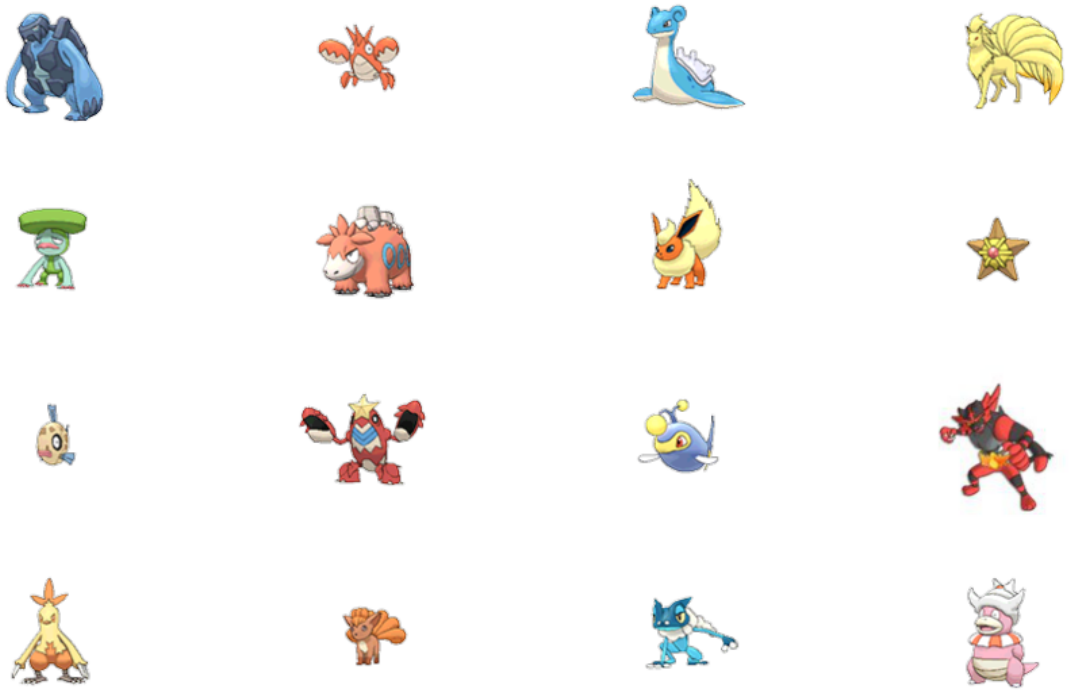
[26]:
```
#shuffle the data
firewater = firewater.sample(frac=1).reset_index(drop=True)

train_gen = keras.preprocessing.image.ImageDataGenerator(
    validation_split=0.2,   # split the dataset into a training set and a␣
 ↪validation set in an 8:2 ratio
    rescale=1./255          # rescale the rgb values to fit between 0 and 1
)
```

[27]:
```
train_data = train_gen.flow_from_dataframe(
    firewater,
    x_col='Filepath',
    y_col='Type1',
    target_size=(120, 120),
    color_mode='rgba',
    class_mode='sparse',
    batch_size=32,
    seed=1,
    subset='training'
)

val_data = train_gen.flow_from_dataframe(
    firewater,
    x_col='Filepath',
    y_col='Type1',
    target_size=(120, 120),
    color_mode='rgba',
    class_mode='sparse',
    batch_size=32,
    seed=1,
    subset='validation'
)
```

```
Found 134 validated image filenames belonging to 2 classes.
Found 33 validated image filenames belonging to 2 classes.
```

[28]:
```
sample_image = train_data.next()[0]

plt.figure(figsize=(16,10))
```

```
for i in range(16):
    plt.subplot(4, 4, i + 1)
    plt.imshow(sample_image[i, :, :, :])
    plt.axis('off')
plt.show()
```



[31]:
```
inputs = layers.Input(shape=(120, 120, 4))

x = layers.Conv2D(filters=64, kernel_size=(9, 9), activation='relu')(inputs)
x = layers.MaxPool2D()(x)

x = layers.Conv2D(filters=128, kernel_size=(9, 9), activation='relu')(x)
x = layers.MaxPool2D()(x)

x = layers.Conv2D(filters=256, kernel_size=(9, 9), activation='relu')(x)
x = layers.MaxPool2D()(x)

x = layers.Flatten()(x)
x = layers.Dense(512, activation='relu')(x)
x = layers.Dropout(0.5)(x)

output = layers.Dense(units=1, activation='sigmoid')(x)
```

```
model = keras.Model(inputs=inputs, outputs=output)

model.compile(
    optimizer='Adam',
    loss='binary_crossentropy',
    metrics=['acc', keras.metrics.AUC()]
)

# print model layers
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 120, 120, 4)]     0

 conv2d (Conv2D)             (None, 112, 112, 64)      20800

 max_pooling2d (MaxPooling2D  (None, 56, 56, 64)       0
 )

 conv2d_1 (Conv2D)           (None, 48, 48, 128)       663680

 max_pooling2d_1 (MaxPooling  (None, 24, 24, 128)      0
 2D)

 conv2d_2 (Conv2D)           (None, 16, 16, 256)       2654464

 max_pooling2d_2 (MaxPooling  (None, 8, 8, 256)        0
 2D)

 flatten (Flatten)           (None, 16384)             0

 dense (Dense)               (None, 512)               8389120

 dropout (Dropout)           (None, 512)               0

 dense_1 (Dense)             (None, 1)                 513

=================================================================
Total params: 11,728,577
Trainable params: 11,728,577
Non-trainable params: 0
_________________________________________________________________
```

[32]:
```
history = model.fit(
    train_data,
    validation_data=val_data,
    epochs=30,
    callbacks=[
        keras.callbacks.EarlyStopping(
            monitor='val_loss',
            patience=3,
            restore_best_weights=True
        ),
        keras.callbacks.ReduceLROnPlateau()
    ]
)
```

```
Epoch 1/30
5/5 [==============================] - 50s 10s/step - loss: 2.2072 - acc: 0.4552
- auc: 0.5347 - val_loss: 0.6908 - val_acc: 0.6364 - val_auc: 0.2893 - lr:
0.0010
Epoch 2/30
5/5 [==============================] - 43s 10s/step - loss: 0.6634 - acc: 0.6866
- auc: 0.4413 - val_loss: 0.6169 - val_acc: 0.6667 - val_auc: 0.6860 - lr:
0.0010
Epoch 3/30
5/5 [==============================] - 43s 10s/step - loss: 0.5865 - acc: 0.6866
- auc: 0.7694 - val_loss: 0.5909 - val_acc: 0.6667 - val_auc: 0.8822 - lr:
0.0010
Epoch 4/30
5/5 [==============================] - 43s 10s/step - loss: 0.5487 - acc: 0.6866
- auc: 0.8496 - val_loss: 0.5290 - val_acc: 0.6667 - val_auc: 0.9153 - lr:
0.0010
Epoch 5/30
5/5 [==============================] - 45s 9s/step - loss: 0.4793 - acc: 0.7090
- auc: 0.9191 - val_loss: 0.5172 - val_acc: 0.6667 - val_auc: 0.9318 - lr:
0.0010
Epoch 6/30
5/5 [==============================] - 43s 10s/step - loss: 0.5007 - acc: 0.7313
- auc: 0.8432 - val_loss: 0.4942 - val_acc: 0.6970 - val_auc: 0.8492 - lr:
0.0010
Epoch 7/30
5/5 [==============================] - 45s 9s/step - loss: 0.4069 - acc: 0.8657
- auc: 0.9339 - val_loss: 0.5734 - val_acc: 0.6667 - val_auc: 0.8636 - lr:
0.0010
Epoch 8/30
5/5 [==============================] - 43s 8s/step - loss: 0.3952 - acc: 0.7985
- auc: 0.8910 - val_loss: 0.4089 - val_acc: 0.7273 - val_auc: 0.8492 - lr:
0.0010
Epoch 9/30
5/5 [==============================] - 43s 10s/step - loss: 0.3399 - acc: 0.8433
```

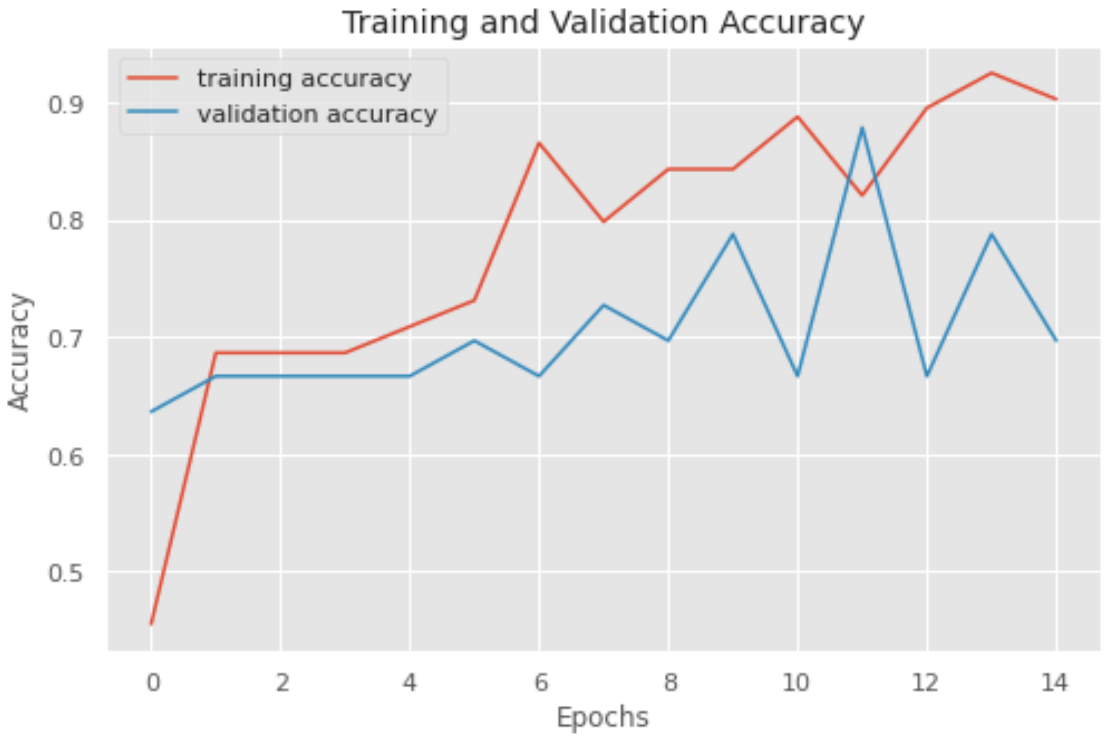
```
- auc: 0.9281 - val_loss: 0.5054 - val_acc: 0.6970 - val_auc: 0.9008 - lr:
0.0010
Epoch 10/30
5/5 [==============================] - 43s 8s/step - loss: 0.3693 - acc: 0.8433
- auc: 0.9068 - val_loss: 0.3921 - val_acc: 0.7879 - val_auc: 0.9029 - lr:
0.0010
Epoch 11/30
5/5 [==============================] - 45s 8s/step - loss: 0.2846 - acc: 0.8881
- auc: 0.9529 - val_loss: 0.6269 - val_acc: 0.6667 - val_auc: 0.9752 - lr:
0.0010
Epoch 12/30
5/5 [==============================] - 44s 9s/step - loss: 0.3667 - acc: 0.8209
- auc: 0.9352 - val_loss: 0.3859 - val_acc: 0.8788 - val_auc: 0.8946 - lr:
0.0010
Epoch 13/30
5/5 [==============================] - 43s 8s/step - loss: 0.2925 - acc: 0.8955
- auc: 0.9491 - val_loss: 0.5587 - val_acc: 0.6667 - val_auc: 0.8719 - lr:
0.0010
Epoch 14/30
5/5 [==============================] - 45s 8s/step - loss: 0.2205 - acc: 0.9254
- auc: 0.9636 - val_loss: 0.4579 - val_acc: 0.7879 - val_auc: 0.9132 - lr:
0.0010
Epoch 15/30
5/5 [==============================] - 45s 9s/step - loss: 0.2027 - acc: 0.9030
- auc: 0.9711 - val_loss: 0.7996 - val_acc: 0.6970 - val_auc: 0.9339 - lr:
0.0010
```

[33]:

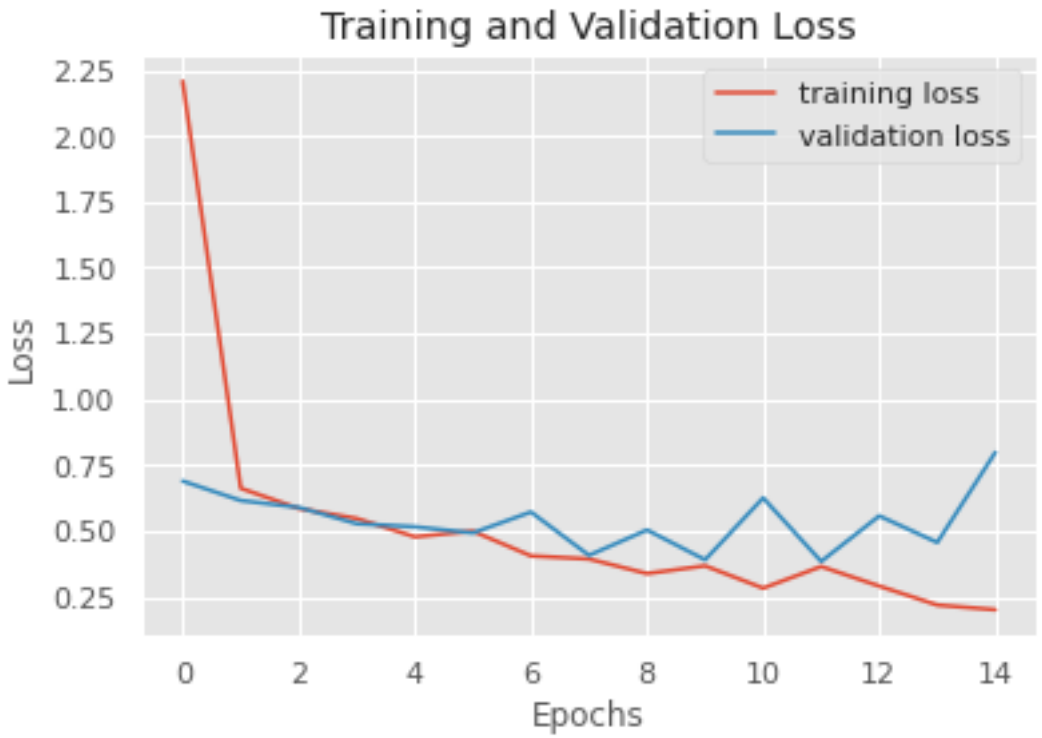
```
plt.style.use('ggplot')

# retrieve accuracy history on training and validation data
acc = history.history['acc']
val_acc = history.history['val_acc']

# retrieve loss history on training and validation data
loss = history.history['loss']
val_loss = history.history['val_loss']

# get number of epochs
epochs = range(len(acc))

plt.figure(figsize=(8, 5))

# plot training and validation accuracy per epoch
plt.plot(epochs, acc, label='training accuracy')
plt.plot(epochs, val_acc, label='validation accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epochs')
```

```
plt.ylabel('Accuracy')
plt.legend()

plt.figure(figsize=(8, 5))

# plot training and validation loss per epoch
plt.figure()
plt.plot(epochs, loss, label='training loss')
plt.plot(epochs, val_loss, label='validation loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
```

[33]: <matplotlib.legend.Legend at 0x7f8785474ed0>



<Figure size 576x360 with 0 Axes>



[34]:
```
# get true labels
true_labels = val_data.labels

# get predictions in the form of probablities
predictions = model.predict(val_data)

# convert probablities into binary values
predictions = [1 if n >= 0.5 else 0 for n in predictions]
print("Model predictions: "+str(predictions))
print("Actual labels:     "+str(true_labels))

# determine filepaths of misclassified pokemon
num_misclasssified = 0
misclassified_filepaths = []
correctness = []
for pred, label, i in zip(predictions, true_labels, range(len(predictions))):
  misclassified_filepaths.append(val_data.filepaths[i])
  if pred != label:
    correctness.append('incorrect')
    num_misclasssified += 1
  else:
    correctness.append('correct')
```

```
print("# of misclassified pokemon: "+str(num_misclasssified))
```

```
Model predictions: [0, 0, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0,
1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0]
Actual labels:     [1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1,
0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1]
# of misclassified pokemon: 12
```

[35]:
```
# obtain the images from the filepath at the determined indices
misclassified_imgs = []
for filepath in misclassified_filepaths:
  misclassified_imgs.append(mpimg.imread(filepath, 0))

# plot results
f, axarr = plt.subplots(6,5, figsize=(20,10))
count = 0
for r in range(6):
  for c in range(5):
    axarr[r,c].imshow(misclassified_imgs[count])
    if correctness[count] == 'correct':
      axarr[r,c].set_title(correctness[count])
    else:
      axarr[r,c].set_title(correctness[count], color='red')
    axarr[r,c].set_axis_off()
    count += 1
plt.show()
```