In [1]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

In [ ]:
```
!pip3 install -U tensorflow_decision_forests
```

In [3]:
```
# standard and PIL Image
import pandas as pd
import numpy as np
import random
import os
from PIL import Image

# tf and keras
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator, array_to_i
from keras import models
from keras import layers
import tensorflow_decision_forests as tfdf

# plots
import seaborn as sns
import matplotlib.pyplot as plt

random.seed(2)
%matplotlib inline

IMAGE_PATH = '/content/drive/MyDrive/w207_final_project/archive/'
```

In [11]:
```
def preprocessing_labels():
    image_values_df = pd.read_csv('/content/drive/MyDrive/w207_final_project/ne
    image_values_df["type"] = image_values_df[["MEL", "NV", "BCC", "AK", "BKL",
    image_values_df = image_values_df[['image', 'type']]
    print("Number of rows in Image data set before Merge:", len(image_values_df
    metadata_df = pd.read_csv('/content/drive/MyDrive/w207_final_project/new_ar
    print("Number of rows in Metadata data set before Merge:", len(metadata_df)
    image_values_df = image_values_df.merge(metadata_df, on = 'image', how = 'i
    print("Number of rows in Merged Dataset:", len(image_values_df))
    return image_values_df
```

In [48]:
```
labels_df = preprocessing_labels()
labels_df
```

```
Number of rows in Image data set before Merge: 25331
Number of rows in Metadata data set before Merge: 25331
Number of rows in Merged Dataset: 25331
```

Out[48]:

	image	type	age_approx	anatom_site_general	lesion_id	sex
0	ISIC_0000000	NV	55.0	anterior torso	NaN	female
1	ISIC_0000001	NV	30.0	anterior torso	NaN	female
2	ISIC_0000002	MEL	60.0	upper extremity	NaN	female
3	ISIC_0000003	NV	30.0	upper extremity	NaN	male
4	ISIC_0000004	MEL	80.0	posterior torso	NaN	male
...	...	...	...	...	...	...
25326	ISIC_0073247	BCC	85.0	head/neck	BCN_0003925	female
25327	ISIC_0073248	BKL	65.0	anterior torso	BCN_0001819	male
25328	ISIC_0073249	MEL	70.0	lower extremity	BCN_0001085	male
25329	ISIC_0073251	NV	55.0	palms/soles	BCN_0002083	female
25330	ISIC_0073254	BKL	50.0	upper extremity	BCN_0001079	male

25331 rows × 6 columns

RANDOM FOREST

In [54]:
```
images = pd.read_csv('/content/drive/MyDrive/w207_final_project/our_images.csv'
labels_df = labels_df[labels_df['image'].isin(list(images['image_id'].values
                                                    ))][['image', 'type', 'age_a
                                                         'anatom_site_general',
images_1 = images['image_id'].isin(labels_df['image'].values)
```

In [57]:
```
labels_df = labels_df.sort_values('image')
labels_df['image_data'] = images[images_1].sort_values('image_id')['images'].va
labels_df
```

Out [57]:

	image	type	age_approx	anatom_site_general	sex	image_data
7	ISIC_0000008	NV	30.0	anterior torso	female	[[[167. 164. 175.]\n [153. 150. 161.]\n [160...
18	ISIC_0000019_downsampled	NV	30.0	posterior torso	female	[[[0. 0. 2.]\n [0. 0. 2.]\n [0...
21	ISIC_0000022_downsampled	MEL	55.0	lower extremity	female	[[[9. 7. 8.]\n [2. 0. 1.]\n [0...
22	ISIC_0000023_downsampled	NV	30.0	anterior torso	female	[[[0. 0. 2.]\n [0. 0. 4.]\n [0...
23	ISIC_0000024_downsampled	NV	45.0	posterior torso	male	[[[0. 0. 2.]\n [0. 1. 3.]\n [1...
...	...	...	...	...	...	...
25322	ISIC_0073241	MEL	60.0	palms/soles	male	[[[7. 5. 6.]\n [11. 6. 10.]\n [13. 12. 1...
25325	ISIC_0073246	BCC	80.0	anterior torso	male	[[[0. 0. 0.]\n [0. 0. 0.]\n [1. 1. 1.]\n .....
25326	ISIC_0073247	BCC	85.0	head/neck	female	[[[0. 0. 0.]\n [0. 0. 0.]\n [0. 0. 0.]\n .....
25327	ISIC_0073248	BKL	65.0	anterior torso	male	[[[87. 83. 84.]\n [87. 82. 86.]\n [91...
25330	ISIC_0073254	BKL	50.0	upper extremity	male	[[[211. 215. 216.]\n [213. 219. 219.]\n [214...

11021 rows × 6 columns

In [59]:

```
is_melanoma = ['MEL' if i == 'MEL' else 'NOT MEL' for i in labels_df['type']]
mel_df = labels_df
mel_df['is_mel'] = is_melanoma
new_mel_df = mel_df[['image', 'age_approx', 'anatom_site_general', 'sex',
        'is_mel', 'image_data']]
new_mel_df
```

Out[59]:

	image	age_approx	anatom_site_general	sex	is_mel	image_da
7	ISIC_0000008	30.0	anterior torso	female	NOT MEL	[[[167. 16 175.]\n [15 150. 161.] [160
18	ISIC_0000019_downsampled	30.0	posterior torso	female	NOT MEL	[[[0. 2.]\n [0. 2.]\n [0
21	ISIC_0000022_downsampled	55.0	lower extremity	female	MEL	[[[9. 7. 8.] [2. 0. 1.]\n 0
22	ISIC_0000023_downsampled	30.0	anterior torso	female	NOT MEL	[[[0. 2.]\n [0. 4.]\n [0
23	ISIC_0000024_downsampled	45.0	posterior torso	male	NOT MEL	[[[0. 2.]\n [0. 3.]\n [1
...	...	...	...	...	...	
25322	ISIC_0073241	60.0	palms/soles	male	MEL	[[[7. 5. 6.] [11. 6. 10.] [13. 12. 1
25325	ISIC_0073246	80.0	anterior torso	male	NOT MEL	[[[0. 0. 0.] [0. 0. 0.] [1. 1. 1.] ..
25326	ISIC_0073247	85.0	head/neck	female	NOT MEL	[[[0. 0. 0.] [0. 0. 0.] [0. 0. 0.] .
25327	ISIC_0073248	65.0	anterior torso	male	NOT MEL	[[[87. 8 84.]\n [8 82. 86.]\n 91
25330	ISIC_0073254	50.0	upper extremity	male	NOT MEL	[[[211. 21 216.]\n [21 219. 219.] [214

11021 rows × 6 columns

In [60]:
```
def df_split(df, split=(0.6,0.2,0.2)):
    """ Split data into train, validation and test sets; apply transformaions a

    Params:
    -------
    images  (np.ndarray): Images of shape (N, 224, 224, 3)
    y (np.ndarray): Labels of shape (N,)
    split (tuple): 3 values summing to 1 defining split of train, validation ar

    Returns:
    --------
    X_train (np.ndarray): Train images of shape (N_train, 224, 224, 3)
```

```
    y_train (np.ndarray): Train labels of shape (N_train,)
    X_val (np.ndarray): Val images of shape (N_val, 224, 224, 3)
    y_val (np.ndarray): Val labels of shape (N_val,)
    X_test (np.ndarray): Test images of shape (N_test, 224, 224, 3)
    y_test (np.ndarray): Test labels of shape (N_test,)

    """

    ### create train/validation/test sets ###
    ##########################################
    # NOTE: Each time you run this cell, you'll re-shuffle the data. The orderi
    splits = np.multiply(len(df), split).astype(int)
    train, val, test = np.split(df.sample(frac = 1), [splits[0], splits[0]+spli
    return train,val,test
```

In [61]:
```
train_df,val_df,test_df = df_split(new_mel_df)
```

In [62]:
```
train_df
```

Out[62]:

	image	age_approx	anatom_site_general	sex	is_mel	image_data
15781	ISIC_0058099	35.0	head/neck	female	NOT MEL	[[[0. 0. 0.]\n [0. 0. 0.]\n [0. 0. 0.]\n .....
18616	ISIC_0062590	75.0	head/neck	male	NOT MEL	[[[141. 130. 128.]\n [143. 132. 128.]\n [143...
20109	ISIC_0065004	70.0	head/neck	male	NOT MEL	[[[0. 0. 0.]\n [0. 0. 0.]\n [0. 0. 0.]\n .....
11330	ISIC_0032733	70.0	posterior torso	male	MEL	[[[0. 0. 0.]\n [0. 0. 0.]\n [0. 0. 0.]\n .....
14835	ISIC_0056571	55.0	lower extremity	female	NOT MEL	[[[45. 39. 39.]\n [50. 42. 40.]\n [53. 44. 4...
...	...	...	...	...	...	...
23789	ISIC_0070809	60.0	anterior torso	male	MEL	[[[5. 5. 5.]\n [7. 7. 7.]\n [8. 8. ...
12010	ISIC_0033413	70.0	head/neck	male	NOT MEL	[[[176. 156. 157.]\n [178. 151. 158.]\n [178...
3751	ISIC_0025154	55.0	upper extremity	male	NOT MEL	[[[162. 131. 137.]\n [165. 135. 137.]\n [166...
20991	ISIC_0066406	85.0	anterior torso	male	NOT MEL	[[[0. 0. 0.]\n [0. 0. 0.]\n [0. 0. 0.]\n .....
18040	ISIC_0061676	45.0	anterior torso	male	NOT MEL	[[[137. 134. 125.]\n [135. 135. 127.]\n [141...

6612 rows × 6 columns

In [63]:
```
train_ds = tfdf.keras.pd_dataframe_to_tf_dataset(train_df, label='is_mel')
test_ds = tfdf.keras.pd_dataframe_to_tf_dataset(test_df, label='is_mel')
```

In [74]:
```
model_1 = tfdf.keras.RandomForestModel(verbose=1)
```

```
Use /tmp/tmpeyi8n5s3 as temporary training directory
```

In [75]:
```
model_1.fit(x = train_ds)
```

```
Reading training dataset...
Training dataset read in 0:00:00.755052. Found 6612 examples.
Training model...
Model trained in 0:00:02.995184
Compiling model...
Model compiled.
```

Out[75]:
```
<keras.callbacks.History at 0x7f5c8416a430>
```

In [76]:
```
model_1.compile(metrics=["accuracy"])
evaluation = model_1.evaluate(test_ds, return_dict=True)
print()

for name, value in evaluation.items():
  print(f"{name}: {value:.4f}")
```

```
3/3 [==============================] - 0s 40ms/step - loss: 0.0000e+00 - accur
acy: 0.7914

loss: 0.0000
accuracy: 0.7914
```

In [71]:
```
tfdf.model_plotter.plot_model_in_colab(model_1, max_depth = 5)
```

Out[71]:

In [72]:
```
model_1.summary()
```

```
Model: "random_forest_model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
=================================================================
Total params: 1
Trainable params: 0
Non-trainable params: 1
_________________________________________________________________
Type: "RANDOM_FOREST"
Task: CLASSIFICATION
Label: "__LABEL"

Input Features (5):
        age_approx
        anatom_site_general
        image
        image_data
        sex

No weights

Variable Importance: MEAN_MIN_DEPTH:
    1.               "image"  9.375218 ################
    2.             "__LABEL"  9.375218 ################
    3.          "image_data"  5.980693 ##########
    4.                 "sex"  5.198797 ########
    5.          "age_approx"  1.960734 ###
    6. "anatom_site_general"  0.000000

Variable Importance: NUM_AS_ROOT:
    1. "anatom_site_general" 300.000000

Variable Importance: NUM_NODES:
    1.          "age_approx" 31728.000000 ################
    2. "anatom_site_general" 10373.000000 ##
    3.          "image_data" 7747.000000
    4.                 "sex" 6192.000000

Variable Importance: SUM_SCORE:
    1.          "age_approx" 64486.777442 ################
    2. "anatom_site_general" 42422.380015 ########
    3.                 "sex" 15738.464060
    4.          "image_data" 15216.359561



Winner takes all: true
Out-of-bag evaluation: accuracy:0.786146 logloss:5.62069
Number of trees: 300
Total number of nodes: 112380

Number of nodes by tree:
Count: 300 Average: 374.6 StdDev: 8.69022
Min: 351 Max: 395 Ignored: 0
----------------------------------------------
[ 351, 353)  1   0.33%   0.33%
[ 353, 355)  1   0.33%   0.67%
[ 355, 357)  4   1.33%   2.00% #
[ 357, 360) 10   3.33%   5.33% ##
```

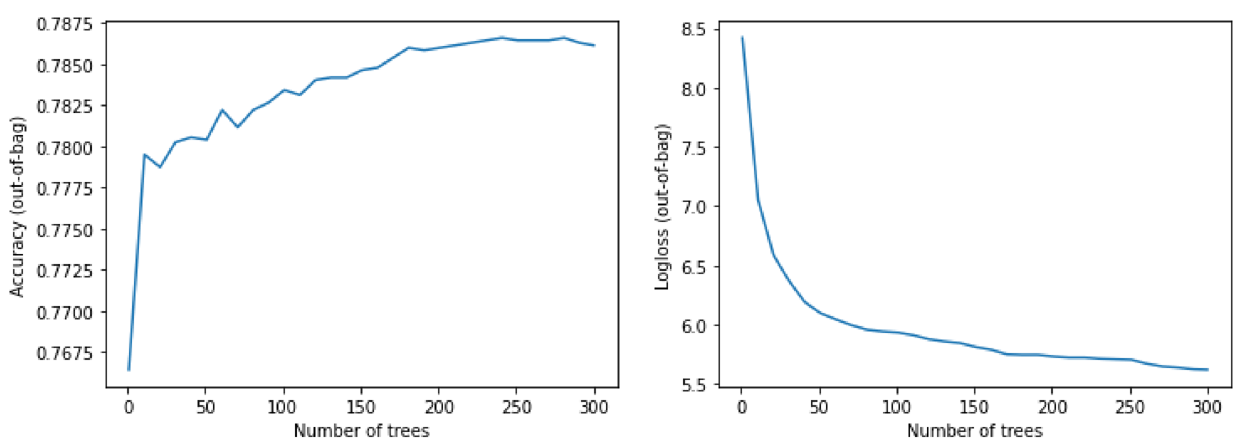
```
[ 360, 362)  8   2.67%   8.00% ##
[ 362, 364) 11   3.67%  11.67% ##
[ 364, 366) 17   5.67%  17.33% ###
[ 366, 369) 10   3.33%  20.67% ##
[ 369, 371) 26   8.67%  29.33% #####
[ 371, 373) 25   8.33%  37.67% #####
[ 373, 375) 30  10.00%  47.67% ######
[ 375, 378) 51  17.00%  64.67% ##########
[ 378, 380) 25   8.33%  73.00% #####
[ 380, 382) 14   4.67%  77.67% ###
[ 382, 384) 19   6.33%  84.00% ####
[ 384, 387) 18   6.00%  90.00% ####
[ 387, 389) 10   3.33%  93.33% ##
[ 389, 391) 12   4.00%  97.33% ##
[ 391, 393)  4   1.33%  98.67% #
[ 393, 395]  4   1.33% 100.00% #

Depth by leafs:
Count: 56340 Average: 9.37526 StdDev: 2.14887
Min: 2 Max: 15 Ignored: 0
----------------------------------------------
[  2,  3)   112   0.20%   0.20%
[  3,  4)   286   0.51%   0.71%
[  4,  5)   645   1.14%   1.85% #
[  5,  6)  1558   2.77%   4.62% #
[  6,  7)  2750   4.88%   9.50% ###
[  7,  8)  4796   8.51%  18.01% ####
[  8,  9)  7584  13.46%  31.47% #######
[  9, 10) 10669  18.94%  50.41% ##########
[ 10, 11) 10920  19.38%  69.79% ##########
[ 11, 12)  8364  14.85%  84.64% ########
[ 12, 13)  5068   9.00%  93.63% #####
[ 13, 14)  2493   4.42%  98.06% ##
[ 14, 15)   869   1.54%  99.60% #
[ 15, 15]   226   0.40% 100.00%

Number of training obs by leaf:
Count: 56340 Average: 35.2077 StdDev: 31.7139
Min: 5 Max: 365 Ignored: 0
----------------------------------------------
[   5,  23) 26356  46.78%  46.78% ##########
[  23,  41) 11740  20.84%  67.62% ####
[  41,  59)  6939  12.32%  79.93% ###
[  59,  77)  5125   9.10%  89.03% ##
[  77,  95)  3073   5.45%  94.49% #
[  95, 113)  1281   2.27%  96.76%
[ 113, 131)   828   1.47%  98.23%
[ 131, 149)   534   0.95%  99.18%
[ 149, 167)   254   0.45%  99.63%
[ 167, 185)   163   0.29%  99.92%
[ 185, 203)    37   0.07%  99.98%
[ 203, 221)     4   0.01%  99.99%
[ 221, 239)     2   0.00%  99.99%
[ 239, 257)     0   0.00%  99.99%
[ 257, 275)     0   0.00%  99.99%
[ 275, 293)     1   0.00%  99.99%
[ 293, 311)     0   0.00%  99.99%
[ 311, 329)     0   0.00%  99.99%
[ 329, 347)     2   0.00% 100.00%
[ 347, 365]     1   0.00% 100.00%
```

```
Attribute in nodes:
        31728 : age_approx [NUMERICAL]
        10373 : anatom_site_general [CATEGORICAL]
        7747 : image_data [CATEGORICAL]
        6192 : sex [CATEGORICAL]

Attribute in nodes with depth <= 0:
        300 : anatom_site_general [CATEGORICAL]

Attribute in nodes with depth <= 1:
        615 : anatom_site_general [CATEGORICAL]
        198 : age_approx [NUMERICAL]
        77 : sex [CATEGORICAL]
        10 : image_data [CATEGORICAL]

Attribute in nodes with depth <= 2:
        969 : age_approx [NUMERICAL]
        689 : anatom_site_general [CATEGORICAL]
        267 : sex [CATEGORICAL]
        63 : image_data [CATEGORICAL]

Attribute in nodes with depth <= 3:
        2238 : age_approx [NUMERICAL]
        881 : anatom_site_general [CATEGORICAL]
        462 : sex [CATEGORICAL]
        297 : image_data [CATEGORICAL]

Attribute in nodes with depth <= 5:
        6433 : age_approx [NUMERICAL]
        2025 : anatom_site_general [CATEGORICAL]
        1772 : image_data [CATEGORICAL]
        1495 : sex [CATEGORICAL]

Condition type in nodes:
        31728 : HigherCondition
        24312 : ContainsBitmapCondition
Condition type in nodes with depth <= 0:
        300 : ContainsBitmapCondition
Condition type in nodes with depth <= 1:
        702 : ContainsBitmapCondition
        198 : HigherCondition
Condition type in nodes with depth <= 2:
        1019 : ContainsBitmapCondition
        969 : HigherCondition
Condition type in nodes with depth <= 3:
        2238 : HigherCondition
        1640 : ContainsBitmapCondition
Condition type in nodes with depth <= 5:
        6433 : HigherCondition
        5292 : ContainsBitmapCondition
Node format: NOT_SET

Training OOB:
        trees: 1, Out-of-bag evaluation: accuracy:0.766409 logloss:8.41949
        trees: 11, Out-of-bag evaluation: accuracy:0.779504 logloss:7.05582
        trees: 21, Out-of-bag evaluation: accuracy:0.778736 logloss:6.5862
        trees: 31, Out-of-bag evaluation: accuracy:0.780248 logloss:6.36837
        trees: 41, Out-of-bag evaluation: accuracy:0.780551 logloss:6.19245
        trees: 51, Out-of-bag evaluation: accuracy:0.780399 logloss:6.09901
```

```
trees: 61, Out-of-bag evaluation: accuracy:0.782214 logloss:6.04538
trees: 71, Out-of-bag evaluation: accuracy:0.781155 logloss:5.99686
trees: 81, Out-of-bag evaluation: accuracy:0.782214 logloss:5.95852
trees: 91, Out-of-bag evaluation: accuracy:0.782668 logloss:5.94392
trees: 101, Out-of-bag evaluation: accuracy:0.783424 logloss:5.93493
trees: 111, Out-of-bag evaluation: accuracy:0.783122 logloss:5.91201
trees: 121, Out-of-bag evaluation: accuracy:0.784029 logloss:5.87801
trees: 131, Out-of-bag evaluation: accuracy:0.78418 logloss:5.85901
trees: 141, Out-of-bag evaluation: accuracy:0.78418 logloss:5.84532
trees: 151, Out-of-bag evaluation: accuracy:0.784634 logloss:5.81237
trees: 161, Out-of-bag evaluation: accuracy:0.784785 logloss:5.78922
trees: 171, Out-of-bag evaluation: accuracy:0.78539 logloss:5.75106
trees: 181, Out-of-bag evaluation: accuracy:0.785995 logloss:5.74651
trees: 191, Out-of-bag evaluation: accuracy:0.785844 logloss:5.74664
trees: 201, Out-of-bag evaluation: accuracy:0.785995 logloss:5.73272
trees: 211, Out-of-bag evaluation: accuracy:0.786146 logloss:5.72387
trees: 221, Out-of-bag evaluation: accuracy:0.786298 logloss:5.72394
trees: 231, Out-of-bag evaluation: accuracy:0.786449 logloss:5.71433
trees: 241, Out-of-bag evaluation: accuracy:0.7866 logloss:5.70988
trees: 251, Out-of-bag evaluation: accuracy:0.786449 logloss:5.70547
trees: 261, Out-of-bag evaluation: accuracy:0.786449 logloss:5.67217
trees: 271, Out-of-bag evaluation: accuracy:0.786449 logloss:5.64847
trees: 281, Out-of-bag evaluation: accuracy:0.7866 logloss:5.63915
trees: 291, Out-of-bag evaluation: accuracy:0.786298 logloss:5.62502
trees: 300, Out-of-bag evaluation: accuracy:0.786146 logloss:5.62069
```

In [73]:

```python
import matplotlib.pyplot as plt

logs = model_1.make_inspector().training_logs()

plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot([log.num_trees for log in logs], [log.evaluation.accuracy for log in l
plt.xlabel("Number of trees")
plt.ylabel("Accuracy (out-of-bag)")

plt.subplot(1, 2, 2)
plt.plot([log.num_trees for log in logs], [log.evaluation.loss for log in logs]
plt.xlabel("Number of trees")
plt.ylabel("Logloss (out-of-bag)")

plt.show()
```



In [1]:

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Cell In [1], line 1
----> 1 import pandoc

ModuleNotFoundError: No module named 'pandoc'
```

In [ ]: