Udacity Machine Learning Engineer Nanodegree

Melanoma Classification - Project Report

Pritesh Shrivastava July 17th, 2020

Definition

Project Overview

Skin cancer is the most prevalent type of cancer. Melanoma, specifically, is responsible for 75% of skin cancer deaths, despite being the least common skin cancer. The American Cancer Society estimates over 100,000 new melanoma cases will be diagnosed in 2020. It's also expected that almost 7,000 people will die from the disease. As with other cancers, early and accurate detection—potentially aided by data science—can make treatment more effective.

Currently, dermatologists evaluate every one of a patient's moles to identify outlier lesions or “ugly ducklings” that are most likely to be melanoma. Existing AI approaches have not adequately considered this clinical frame of reference. Dermatologists could enhance their diagnostic accuracy if detection algorithms take into account “contextual” images within the same patient to determine which images represent a melanoma. If successful, classifiers would be more accurate and could better support dermatological clinic work.

As the leading healthcare organization for informatics in medical imaging, the Society for Imaging Informatics in Medicine (SIIM)'s mission is to advance medical imaging informatics through education, research, and innovation in a multi-disciplinary community. SIIM is joined by the International Skin Imaging Collaboration (ISIC), an international effort to improve melanoma diagnosis. The ISIC Archive contains the largest publicly available collection of quality-controlled dermoscopic images of skin lesions. This dataset has been relesed on the competitive data science platform Kaggle.

Source

Problem Statement

In this competition, participants will identify melanoma in images of skin lesions. In particular, they will use images within the same patient and determine which are likely to represent a melanoma. Specifically, participants need to predict a binary target for each image ie, the probability (floating point) between 0.0 and 1.0 that the lesion in the image is malignant (the target).

Solution Stratgy

For this competition, we are going to build an image classifier using deep learning. We will need to begin with image pre-processing as we have images of varying sizes, for eg., 1024x1024x3 vs 512x512x3 etc. We can combine the results of the image classifier with a tabular data model on image metadata for ensembling.

We can use stratified k-folds for model validation before making predictions on the test set. Since training a deep learning model on a large image dataset (~120 GB) is going to be a compute heavy task, Kaggle notebooks which offer free GPUs (and TPUs) can serve as the ideal solution for training this model. Additionally, pretrained models such as ImageNet might be explored to get a good score.

Metrics

Evaluation metric for this Kaggle competition is Area under the ROC curve. This value is calculated on the test set. As the competition is currently open, its public leaderboard is calculated with approximately 30% of the test data. The final results will be based on the other 70 % of test data. We will try to optimize our AUROC value on the public leaderboard for this capstone project.

An ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters:

- True Positive Rate (TPR), or Recall, is defined as follows :

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

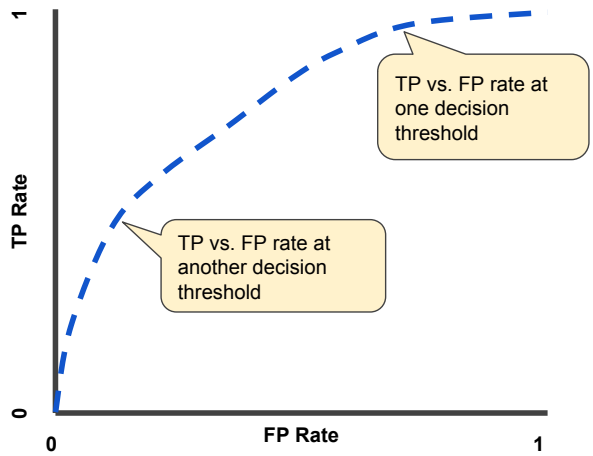
where TP = True Positives & FN = False Negtives

- False Positive Rate (FPR) is defined as follows :

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

where, FP = False Positives & TN = True Negatives

An ROC curve plots TPR vs. FPR at different classification thresholds. Lowering the classification threshold classifies more items as positive, thus increasing both False Positives and True Positives.



Source: https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc

AUC is desirable for the following two reasons:

- AUC is scale-invariant. It measures how well predictions are ranked, rather than their absolute values.
- AUC is classification-threshold-invariant. It measures the quality of the model's predictions irrespective of what classification threshold is chosen.

Analysis

Data Exploration

The dataset consists of images in :

- DICOM format
- JPEG format in JPEG directory
- TFRecord format in tfrecords directory

Additionally, there is a metadata comprising of train, test and submission file in CSV format.

File Sizes

Total size of the dataset (Images + Files) - 108.19 GB

The sizes of the CSV files are shown below :

- train.csv - 1.96 MB - (33126 records, 8 columns)
- test.csv - 479 KB - (10982 records, 5 columns)
- sample_submission.csv - 161 KB (10982 records, 2 columns)

Column description

The description for columns in these 3 files are :

- image_name - unique identifier, points to filename of related DICOM image
- patient_id - unique patient identifier
- sex - the sex of the patient (when unknown, will be blank)
- age_approx - approximate patient age at time of imaging
- anatom_site_general_challenge - location of imaged site
- diagnosis - detailed diagnosis information (train only)
- benign_malignant - indicator of malignancy of imaged lesion
- target - binarized version of the target variable, the value 0 denotes `benign`, and 1 indicates `malignant`

Sample values from train & test sets

The first 5 records from the 2 CSV files are shown below :

- Train CSV :

	image_name	patient_id	sex	age_approx	anatom_site_general_challenge	diagnosis	benign_malignant	target
0	ISIC_2637011	IP_7279968	male	45.0	head/neck	unknown	benign	0
1	ISIC_0015719	IP_3075186	female	45.0	upper extremity	unknown	benign	0
2	ISIC_0052212	IP_2842074	female	50.0	lower extremity	nevus	benign	0
3	ISIC_0068279	IP_6890425	female	45.0	head/neck	unknown	benign	0
4	ISIC_0074268	IP_8723313	female	55.0	upper extremity	unknown	benign	0

- Test CSV :

	image_name	patient_id	sex	age_approx	anatom_site_general_challenge
0	ISIC_0052060	IP_3579794	male	70.0	NaN
1	ISIC_0052349	IP_7782715	male	40.0	lower extremity
2	ISIC_0058510	IP_7960270	female	55.0	torso
3	ISIC_0073313	IP_6375035	female	50.0	torso
4	ISIC_0073502	IP_0589375	female	45.0	lower extremity

Missing Values

Null value count for both test & train datasets is shown below :

```
Train Set
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33126 entries, 0 to 33125
Data columns (total 8 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   image_name                    33126 non-null  object
 1   patient_id                    33126 non-null  object
 2   sex                           33061 non-null  object
 3   age_approx                    33058 non-null  float64
 4   anatom_site_general_challenge  32599 non-null  object
 5   diagnosis                     33126 non-null  object
 6   benign_malignant              33126 non-null  object
 7   target                        33126 non-null  int64
dtypes: float64(1), int64(1), object(6)
memory usage: 2.0+ MB
None
-------------
Test Set
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10982 entries, 0 to 10981
Data columns (total 5 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   image_name                    10982 non-null  object
 1   patient_id                    10982 non-null  object
 2   sex                           10982 non-null  object
 3   age_approx                    10982 non-null  float64
 4   anatom_site_general_challenge  10631 non-null  object
dtypes: float64(1), object(4)
memory usage: 429.1+ KB
None
```

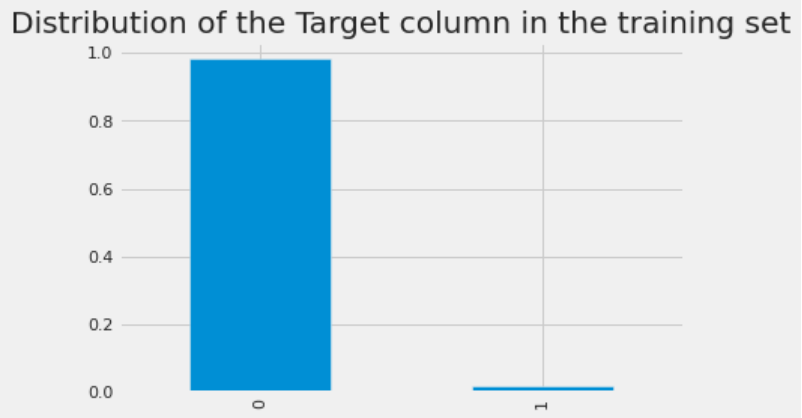
Imputing missing values

We can impute missing values for the categorical variables `sex` & `anotom_site_general_challenge` with `unknown`.

For `age_approx`, we can replace missing values with the mode value.

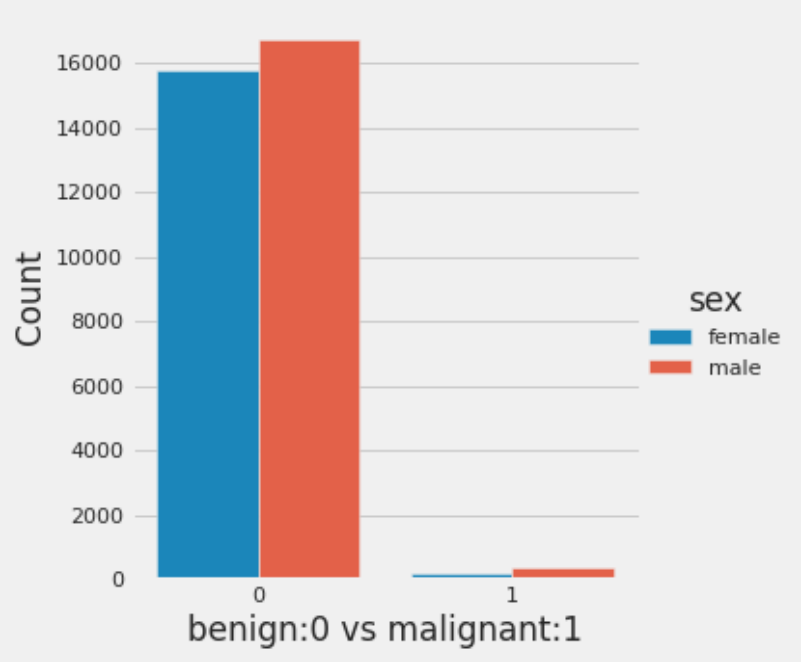
Exploratory Visualization

Target distribution



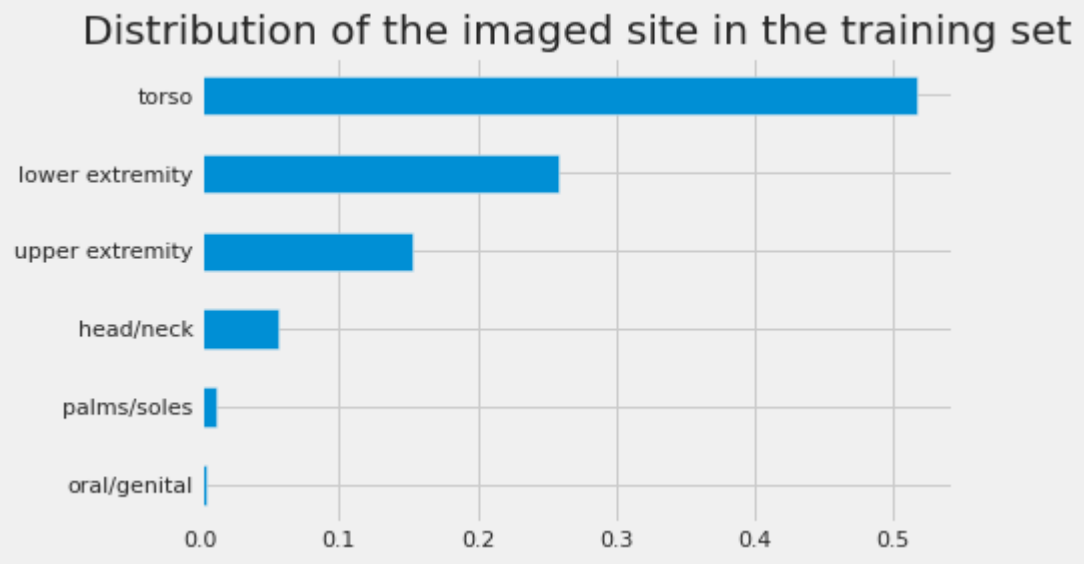
From the given distribution, we can see there is a heavy class imbalance between the benign & malignant cases in the training dataset. For our image classifier, we can perhaps undersample the training data first.

Target distribution by gender

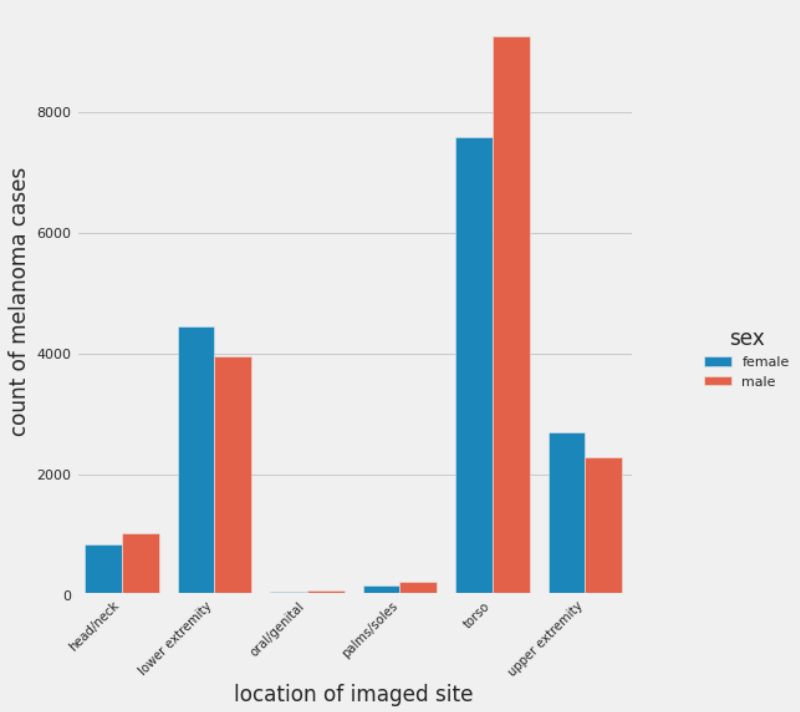


There is a slight difference for benign & malignant cases based on gender.

Distribution of imaged site in training data

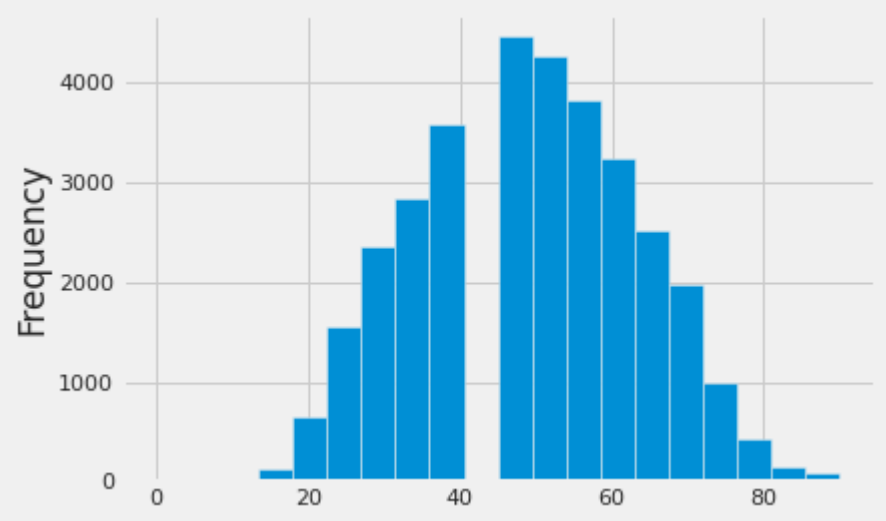


Most of the images have been taken from the torso & lower extremities.

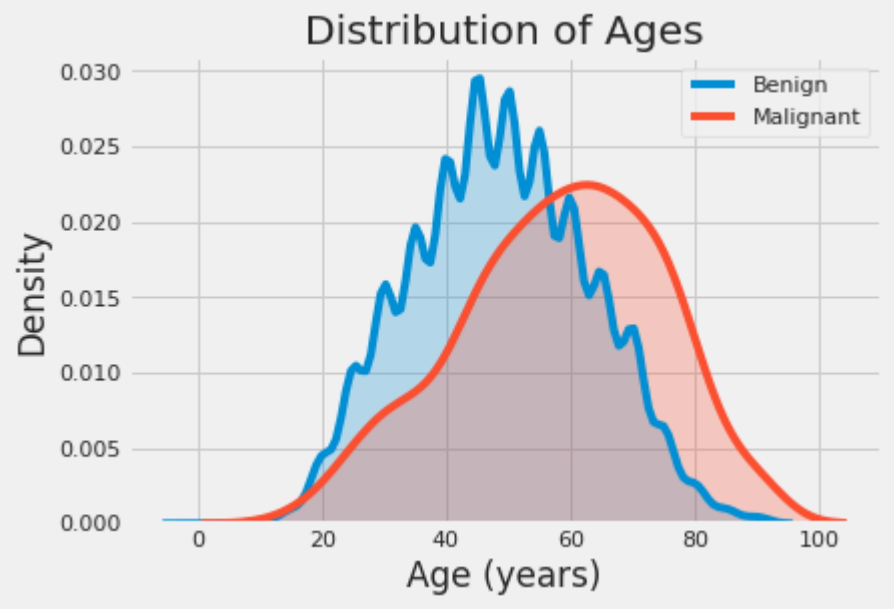


For both genders, the distribution of the imaged sites seem similar

Distribution of patients by age

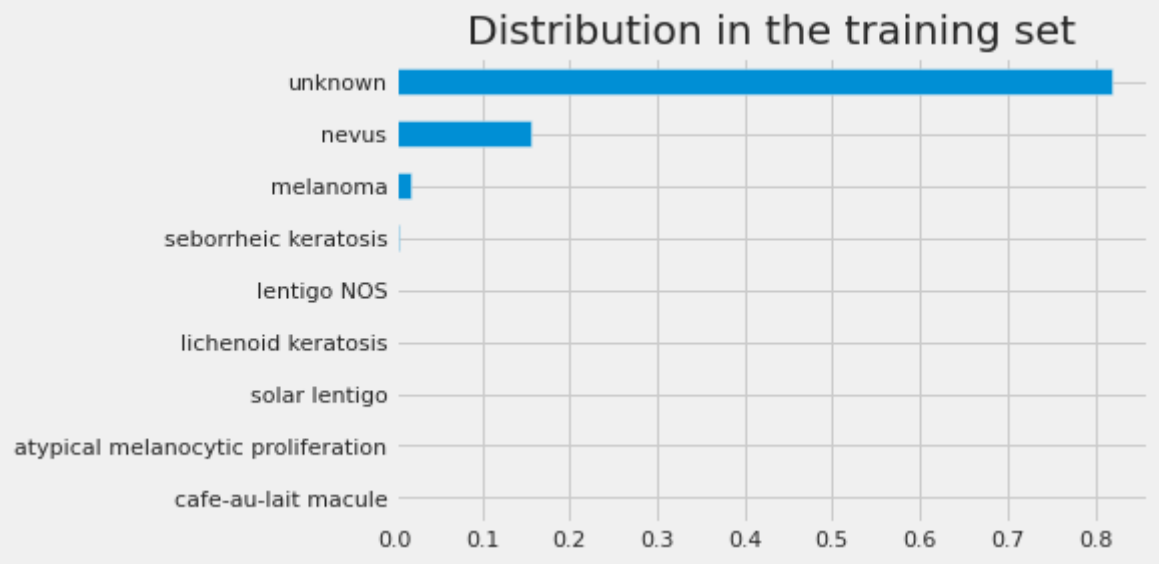


Age distribution of patients seems to follow a normal distribution other than a missing bin near the mode. Note that these are approximate ages though.



There is a higher chance of a malignant cancer in older ages !

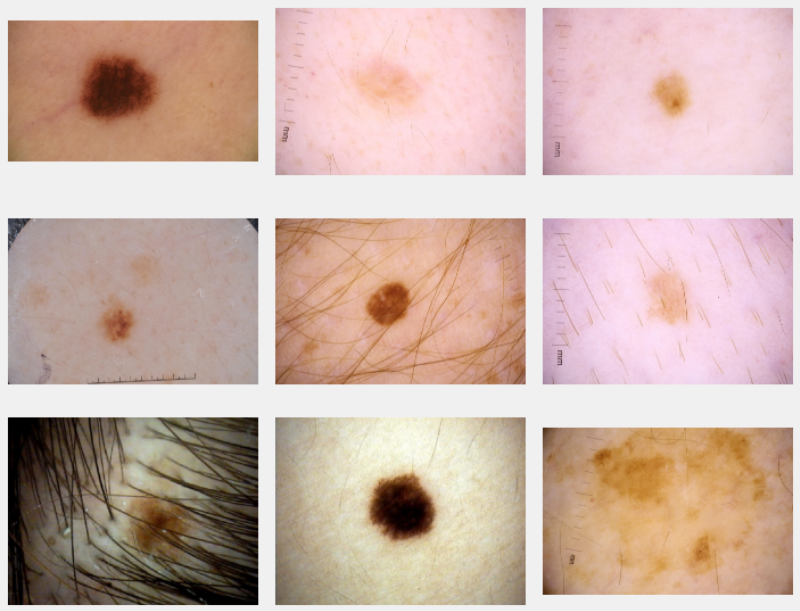
Distribution of manual diagnosis



Most of the cases were diagnosed by medical professionals as either `unknown` or `nevus` (mole).

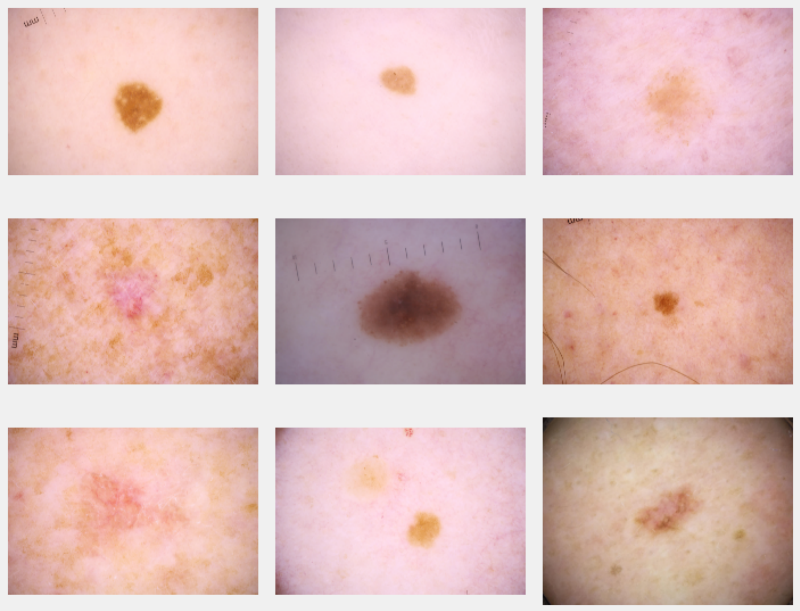
We also checked if the same patient appeared in both the training & test set, but this was not the case.

Sample images

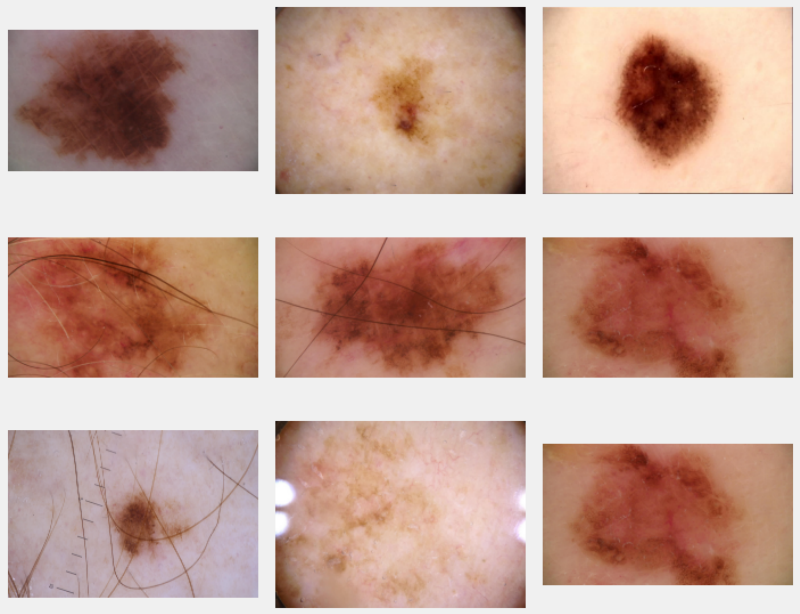


We can see that images have varying sizes, so we will need to resize all images into 1 size before modeling.

Benign cases - Sample images

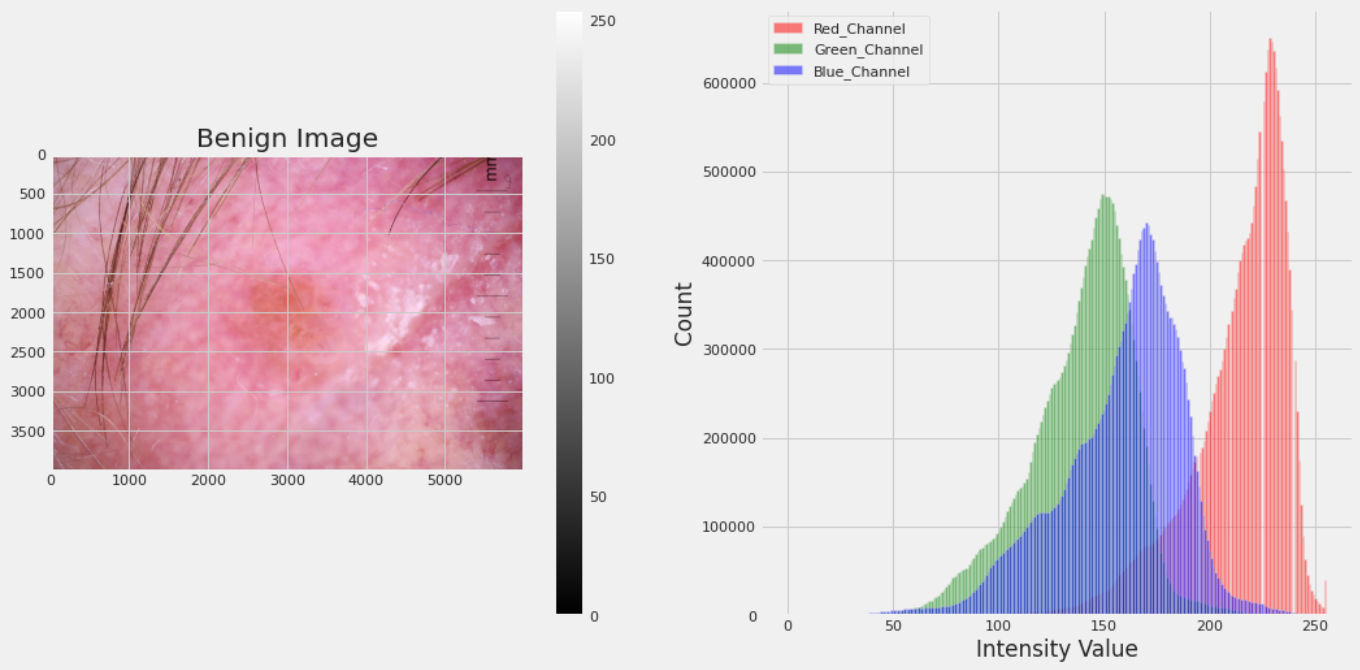


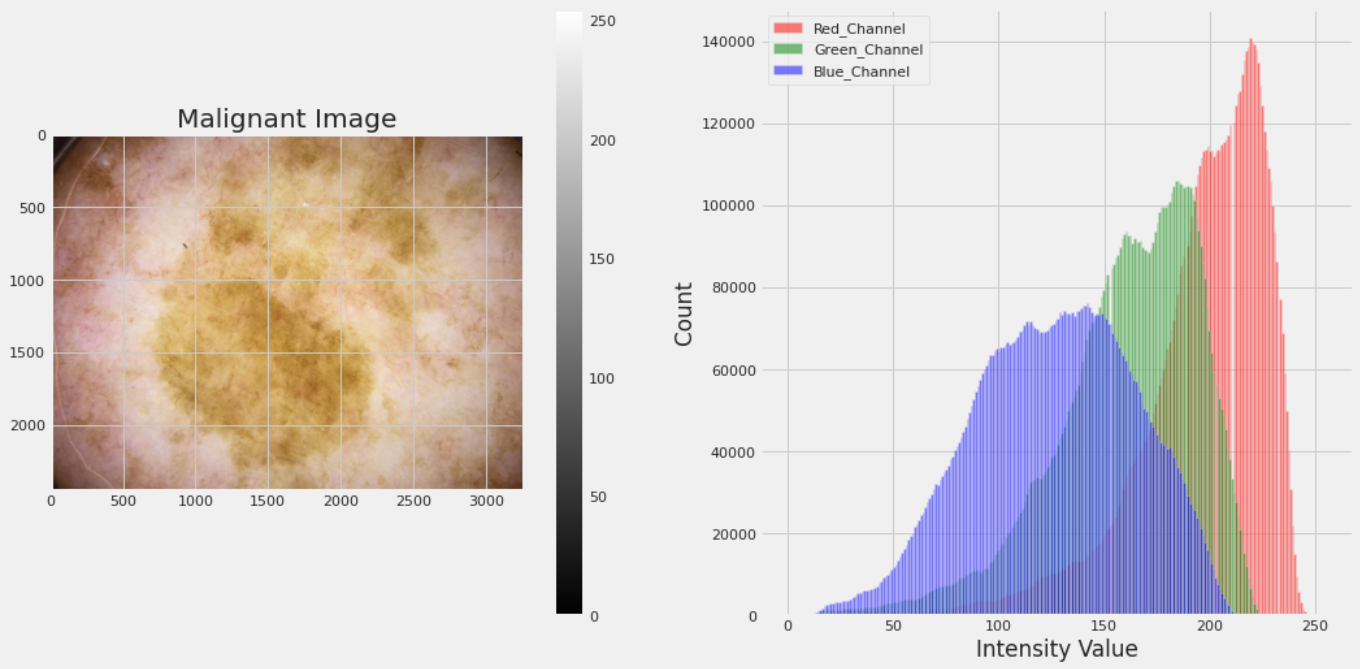
Malignant cases - Sample images



We can see that colors for bening & malignant cases might have different distributions.

Colour distribution - Benign vs Malignant cases





The code for all the above visualizations can be found at `melanoma-eda.ipynb`.

Algorithms and Techniques

For image classification, we are going to use transfer learning. We will use a pre-trained model (VGG16) and modify the output layer using our data. Transfer learning will be helpful in this case because the pre-trained model will already have some kind of intelligence on real world objects built into it which will help with classifying our skin cancer related images as well.

The VGG network is described below :

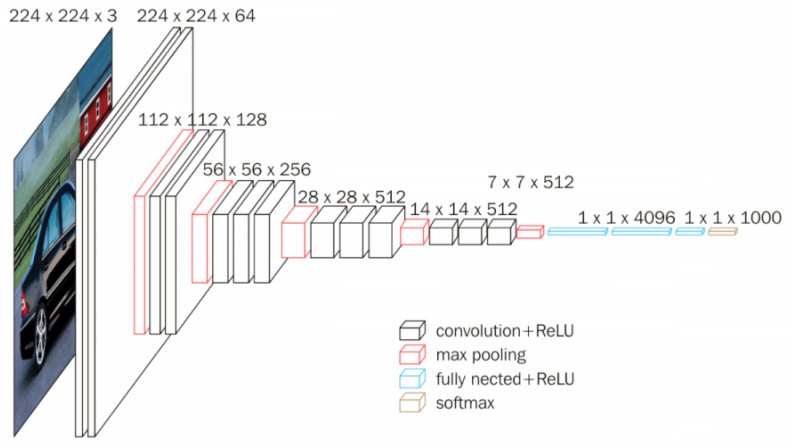
VGG Net - Introduction

The full name of VGG is the Visual Geometry Group, which belongs to the Department of Science and Engineering of Oxford University. It has released a series of convolutional network models beginning with

VGG, which can be applied to face recognition and image classification, from VGG16 to VGG19. The original purpose of VGG's research on the depth of convolutional networks is to understand how the depth of convolutional networks affects the accuracy and accuracy of large-scale image classification and recognition. -Deep-16 CNN), in order to deepen the number of network layers and to avoid too many parameters, a small 3x3 convolution kernel is used in all layers.

Network architecture

The input of VGG is set to an RGB image of 224x244 size. The average RGB value is calculated for all images on the training set image, and then the image is input as an input to the VGG convolution network. A 3x3 or 1x1 filter is used, and the convolution step is fixed. . There are 3 VGG fully connected layers, which can vary from VGG11 to VGG19 according to the total number of convolutional layers + fully connected layers. The minimum VGG11 has 8 convolutional layers and 3 fully connected layers. The maximum VGG19 has 16 convolutional layers. +3 fully connected layers. In addition, the VGG network is not followed by a pooling layer behind each convolutional layer, or a total of 5 pooling layers distributed under different convolutional layers. The following figure is VGG Structure diagram:



Source : https://neurohive.io/en/popular-networks/vgg16/

VGG16

VGG16 contains 16 layers and VGG19 contains 19 layers. A series of VGGs are exactly the same in the last three fully connected layers. The overall structure includes 5 sets of convolutional layers, followed by a MaxPool. The difference is that more and more cascaded convolutional layers are included in the five sets of convolutional layers.

Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

Source : http://ethereon.github.io/netscope/#/preset/vgg-16

XGBoost for Tabular Data

For the image metadata, we are going to use an XGBoost model as these models have traditionally given best possible results on tabular data. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way.

XGBoost stands for “Extreme Gradient Boosting”, where the term “Gradient Boosting” originates from the paper Greedy Function Approximation: A Gradient Boosting Machine, by Friedman. In boosting, the decision trees are built sequentially such that each subsequent tree aims to reduce the errors of the previous tree. Each tree learns from its predecessors and updates the residual errors. Hence, the tree that grows next in the sequence will learn from an updated version of the residuals.

The base learners in boosting are weak learners in which the bias is high, and the predictive power is just a tad better than random guessing. Each of these weak learners contributes some vital information for prediction, enabling the boosting technique to produce a strong learner by effectively combining these weak learners. The final strong learner brings down both the bias and the variance.

The boosting ensemble technique consists of three simple steps:

- An initial model `F0` is defined to predict the target variable `y`. This model will be associated with a residual `(y – F0)`
- A new model `h1` is fit to the residuals from the previous step

- Now, `F0` and `h1` are combined to give `F1`, the boosted version of `F0`. The mean squared error from F1 will be lower than that from `F0`
- To improve the performance of `F1`, we could model after the residuals of `F1` and create a new model `F2`. This can be done for n iterations, until residuals have been minimized as much as possible.

Source

Benchmark

A good baseline model can be created by using 3 features, namely, age, sex and the location of the images site. We can calculate the grouped mean value for each combination of these features in the train set and use that to make predictions on the test set.

Predictions using this simple mean value of the target variable gives an Area under ROC value of `0.699` on the test set (public leaderboard) !!

Our final classifier should be able to beat atleast this benchmark to be deemed useful.

The code for the baseline model is added in the notebook `melanoma-simple-baseline.ipynb`.

Methodology

Data Preprocessing

Pre-processing for Image Data

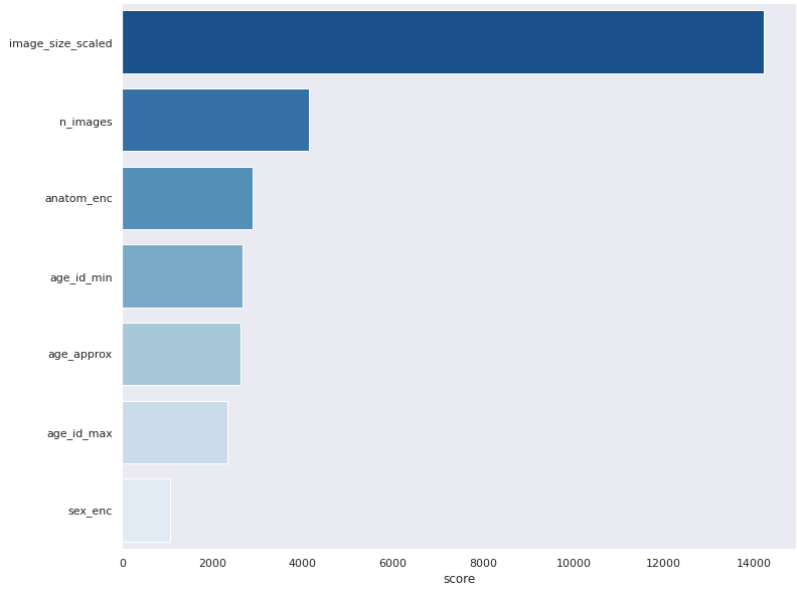
- Under Sampling - Given the biased nature of training data (32542 benign records vs 584 malignant records), we are going to undersample our training set by picking random 2000 record from the benign cases, and all records from the malignant cases.
- Reshaping images - Since the images in our dataset have various sizes, we are going to reshape them all to one size (224x224)
- Train Validation Split - We will create a hold out validation set (20% of train data) to test our model performance before using it on the test set.
- Data Augmentation - We will augment our training data (after exluding the validation data) using basic augmentation techniques like rotation, flipping, shifting height & width, etc.

Code for these pre-processing steps for images is available in the notebook, `melanoma-keras-vgg.ipynb`.

Pre-processing for Tabular Data

- Handling missing values for `sex` & `anotom_site_general_challenge` (imputing with `unknown`) & for `age_approx` (imputing with the mode) has been discussed in the Data Exploration section.
- Label encoding - For categorical variables like `sex` & `anotom_site_general_challenge`, label encoding can be useful.

- Feature Engineering - Grouping `age_approx` by `patient_id`, we can calculate the minimum age, `age_id_min`, and the maximum age, `age_id_max`, that is approximated for each `patient_id`. Besides, I've also added 2 other features related to images - image size (`image_size_scaled`) & no of images per patient (`n_images`). These 2 features may not be applicable in a real world application for melanoma detection, but for the purposes of this competition, I've found that these 2 features have significant feature importance.



Code for tabular data pre-processing can be found in the notebook, `melanoma-tabular-data-xgboost.ipynb`.

Implementation

Image Data Classifier

After data processing of images, we use a pre-trained VGG-16 model trained `imagenet` using the Keras & Tensorflow libraries. We add 1 Dense layer with a sigmoid activation to the pre-trained model without the top layer. For the loss function, we have chosen focal loss rather than binary cross-entropy owing to the class imbalance. Source

We have used an Adam optimizer with a learning rate of `1e-5` & a batch size of 8.

Tabular Data Regressor

As we need to predict a continuous probability between 0 & 1 (float), we are using an XGBoost Regressor model. After the pre-processing & feature engineering, we have manually tuned the parameters of the XGBoost model by using a stratified 5-fold cross validation strategy.

The no of estimators for the regressor were set at 700 with a max_depth of 10. Various hyperparameters were tuned by hand to give the best possible result on public leaderboard.

The training & prediction code for image classifier is available in the notebook `melanoma-keras-vgg.ipynb`, while that for the tabular data regressor is in the notebook `melanoma-tabular-data-xgboost.ipynb`.

Refinement

We began with a simple baseline model that used grouped means to calculate target values. This gave us a score of 69.9% AUROC (public leaderboard).

Before trying an image classifier, we used only the tabular data with an XGBoost model to get an improved result of 72.88 %.

Next, we built an image classifier using transfer learning, with VGG-16 pretrained model, to get a much better result of AUROC 81.16 %.

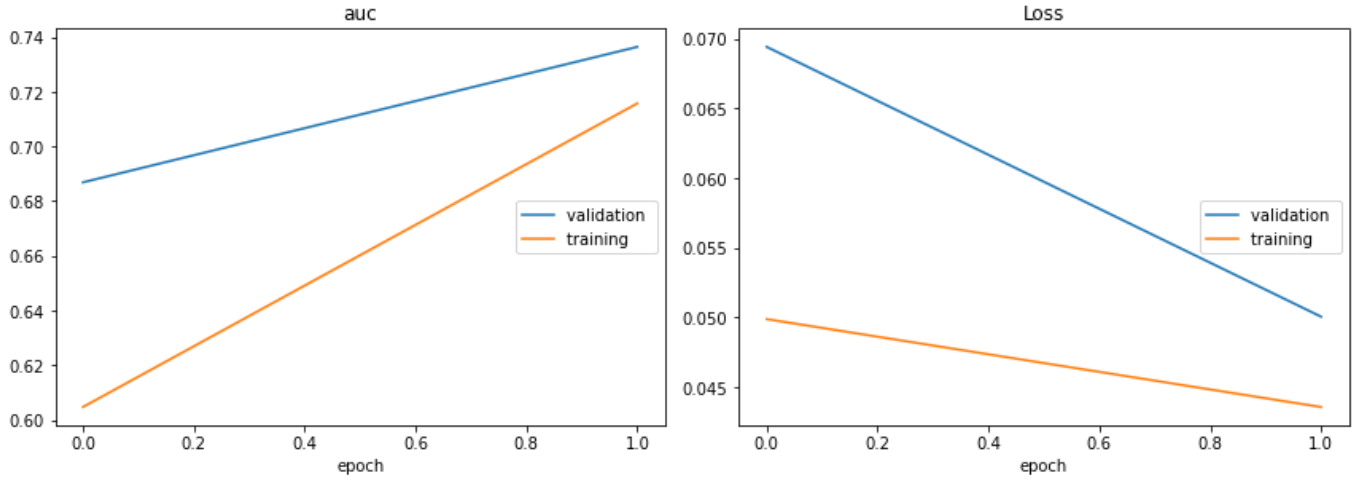
We further tried to improve this result by creating an ensemble of both tablular data (XGBoost) & image data predictions to end up with a result of AUROC 82.39 %.

Results

Model Evaluation and Validation

Model Validation for Image Classifier

We have used a 20% holdout validation set for this case. The results on validation set are shown below :



We are getting a max value of AUROC of 73.6 %. The results seem to improve with no of epochs as the validation loas also reduces.

When using this model on the test set, we get a score of 81.16 % on the public leaderboard (30% of the test set).

Model Validation for Tabular Data Model

Using stratified k-fold (5 folds) for the XGBoost regressor model, we acheived an AUROC of 83.39 %. Fitting the same model on the entire training dat & making predictions on the test set gave an AUROC of 72.88 % on the public leaderboard.

Test set results for the ensemble model

On the public leaderboard, the ensemble of tabular data model & the image classifier gave a better result that either of those models alone, an AUROC of 82.39 %.

Justification

Our ensemble model acheives an Area under ROC curve of 82.39 % on the test set (public leaderboard). This is much better than the resul1ts of our simple baseline model that gave a score of 69.9 %. We have been able to make significant improvement over our evaluation metric through this project !

Future steps for improvement

Use of external data, such as similar image classification competitions from previous years (ISIC 2018, ISIC 2017, ISIC 2016) can be explored.

More sophisticated data augmentation techniques can be tried out.

Features related to the colour of the images can be included in the tabular data classifier to check if it is significant.