Week 3 Project: Mammogram Analysis

Sreeja Challa, Jun Han, Daniel Kim, Angel Li

Table of Contents

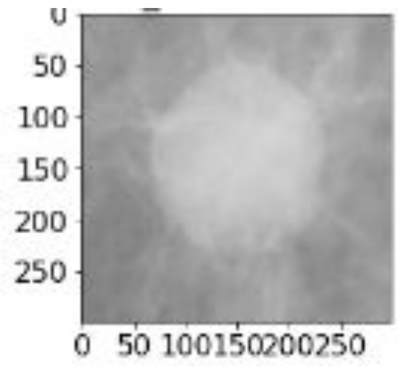
1. Introduction
2. Preprocessing
3. Data Augmentation
    a. Geometric transformations
    b. CLAHE
    c. Jitter
4. Our Models and Hyperparameters
5. Results
    a. CNN model
    b. Random forest
    c. K-means
6. Conclusion

Introduction

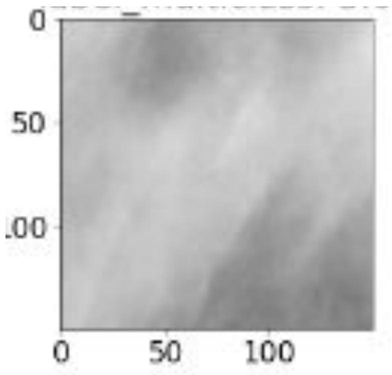
A mammogram is a X-ray that detects signs of breast cancer from abnormal masses and calcifications. This week's challenge is to classify mammogram into two classes (binary) and five classes (multiclass).

Preprocessing

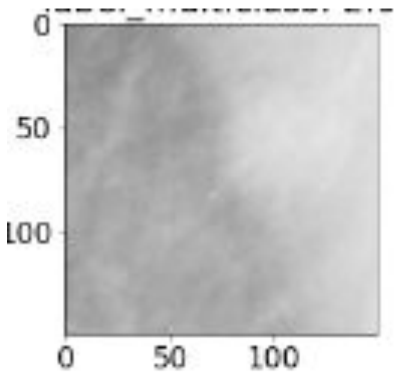
Original



Resize

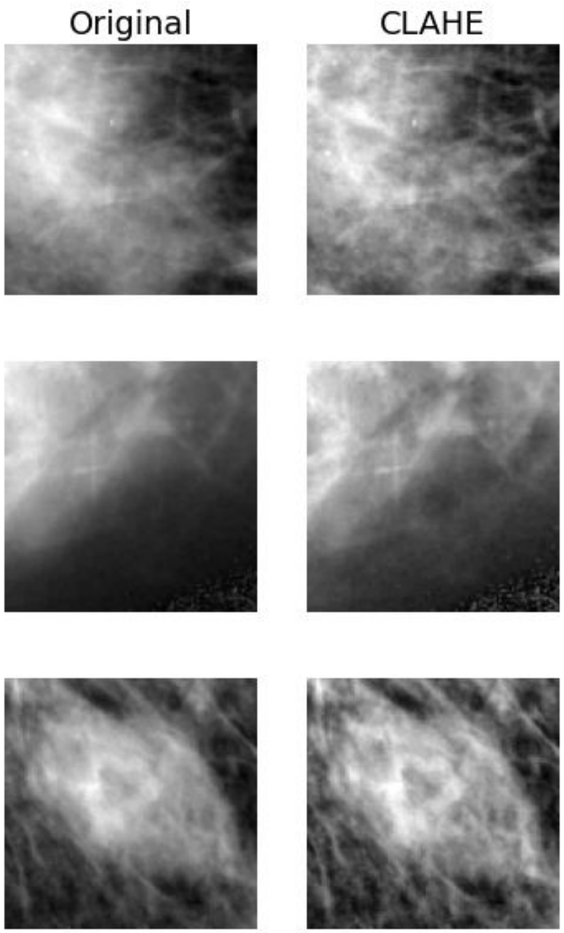


Rescale



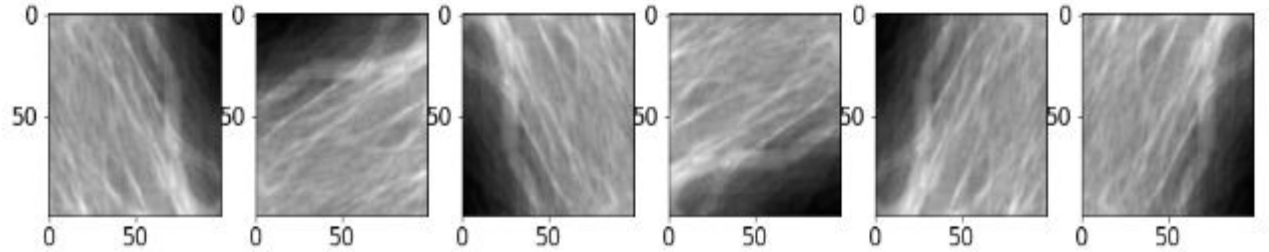
CLAHE

- Contrast limited adaptive histogram equalization
- Improves visibility/contrast
- Contrast applied locally instead of globally



Geometric Transformations

- Augmenting data by rotating and flipping images



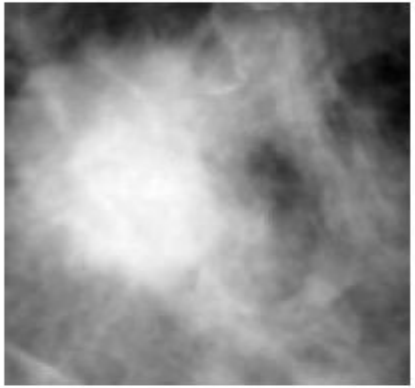
Original 90 degrees 180 degrees 270 degrees Horizontal flip Vertical flip

- Tumors can appear in any orientation

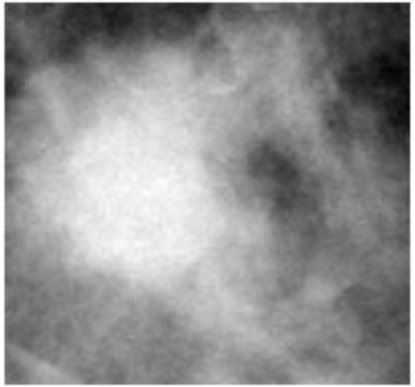
Jitter

- Another augmentation technique
- Adds noise to the entire image
  - Shifts each brightness level by a small random amount
- Improves generalizability

Original



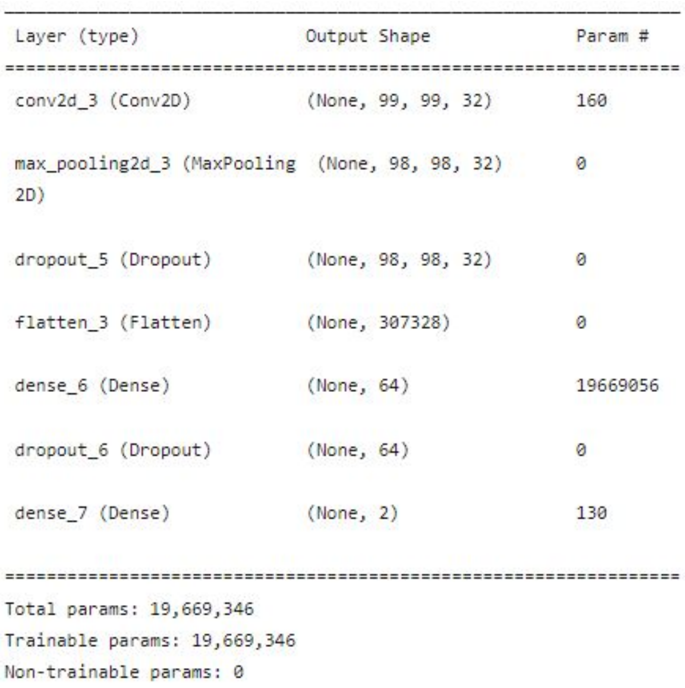
Jitter



Models

Convolutional Neural Network (CNN) for Binary

- Learning rate = 0.001
- Epoch = 10
- Batch size = 64
- Dropout rate = 0.50
- Number of filters = 32

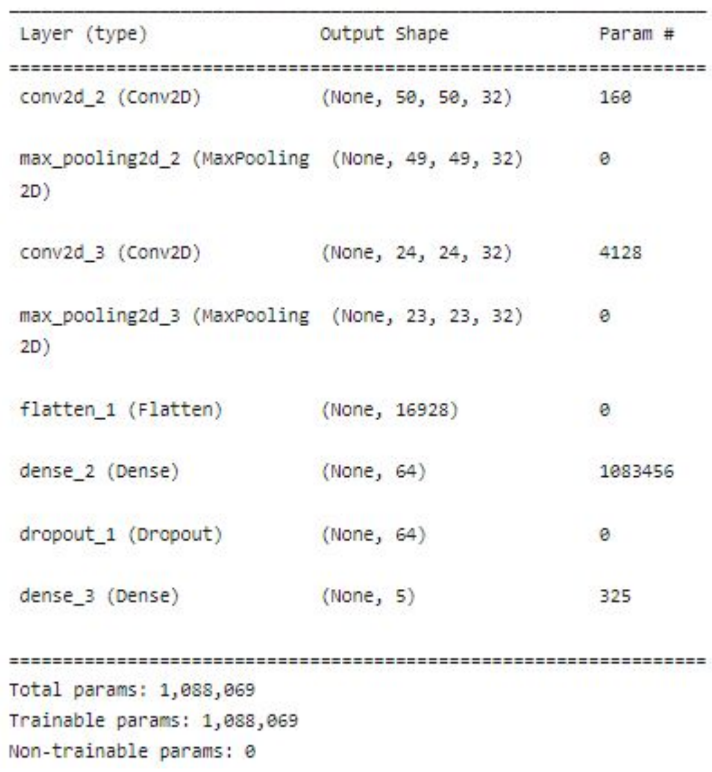


```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_3 (Conv2D)           (None, 99, 99, 32)        160

 max_pooling2d_3 (MaxPooling  (None, 98, 98, 32)       0
 2D)

 dropout_5 (Dropout)         (None, 98, 98, 32)        0

 flatten_3 (Flatten)         (None, 307328)            0

 dense_6 (Dense)             (None, 64)                19669056

 dropout_6 (Dropout)         (None, 64)                0

 dense_7 (Dense)             (None, 2)                 130

=================================================================
Total params: 19,669,346
Trainable params: 19,669,346
Non-trainable params: 0
```

Models

Convolutional Neural Network (CNN) for Multi

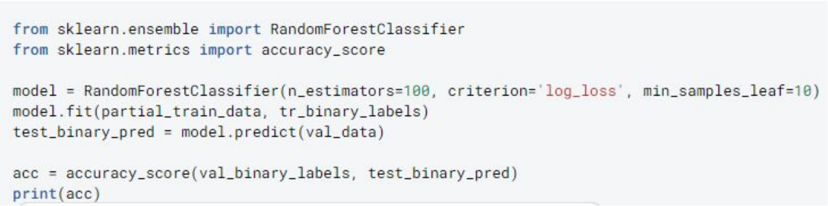
- Learning rate = 0.001
- Epoch = 10
- Batch size = 64
- Dropout rate = 0.50
- Number of filters = 64



```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 50, 50, 32)        160

max_pooling2d_2 (MaxPooling  (None, 49, 49, 32)        0
2D)

conv2d_3 (Conv2D)            (None, 24, 24, 32)        4128

max_pooling2d_3 (MaxPooling  (None, 23, 23, 32)        0
2D)

flatten_1 (Flatten)          (None, 16928)             0

dense_2 (Dense)              (None, 64)                1083456

dropout_1 (Dropout)          (None, 64)                0

dense_3 (Dense)              (None, 5)                 325

=================================================================
Total params: 1,088,069
Trainable params: 1,088,069
Non-trainable params: 0
```

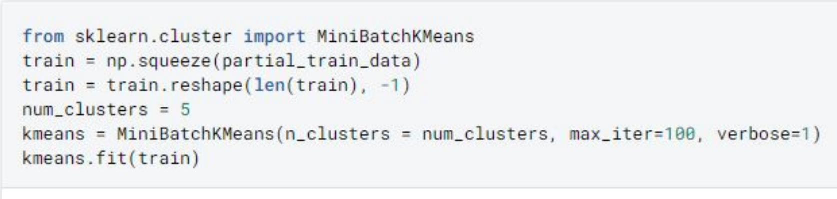
Models

RandomForest



```python
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

model = RandomForestClassifier(n_estimators=100, criterion='log_loss', min_samples_leaf=10)
model.fit(partial_train_data, tr_binary_labels)
test_binary_pred = model.predict(val_data)

acc = accuracy_score(val_binary_labels, test_binary_pred)
print(acc)
```

K-means algorithm



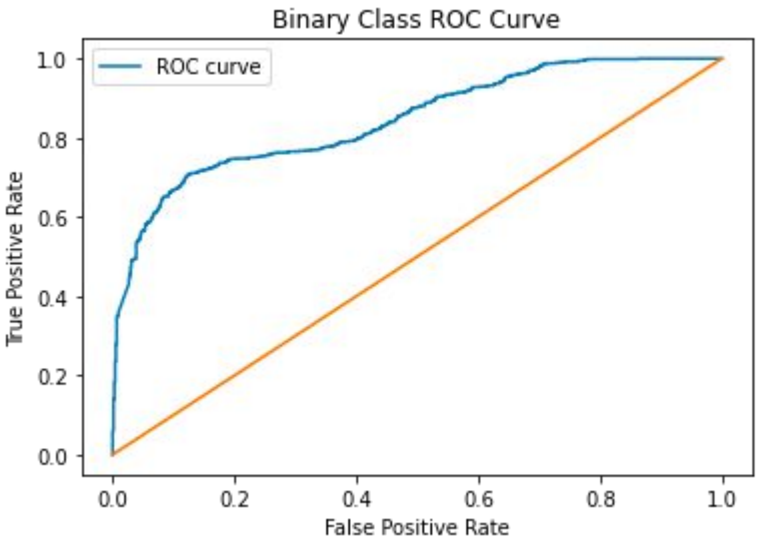
```python
from sklearn.cluster import MiniBatchKMeans
train = np.squeeze(partial_train_data)
train = train.reshape(len(train), -1)
num_clusters = 5
kmeans = MiniBatchKMeans(n_clusters = num_clusters, max_iter=100, verbose=1)
kmeans.fit(train)
```

Results - Binary with CNN

AUC_ROC: 0.8453

Confusion Matrix score: 421

Accuracy: 68.5333

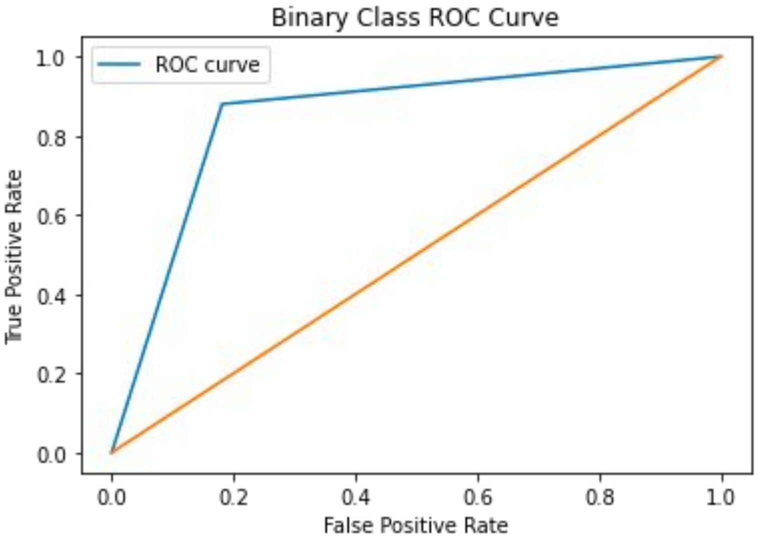


Results - Binary with RandomForest

AUC_ROC: 0.8493

Confusion Matrix score: 1600

Accuracy: 84.9333

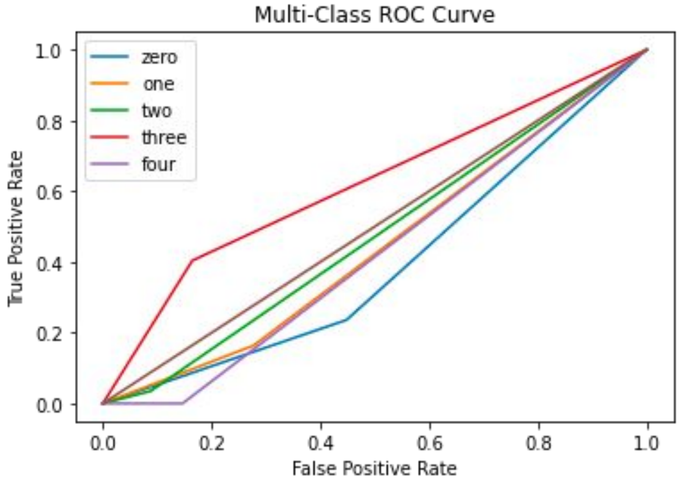


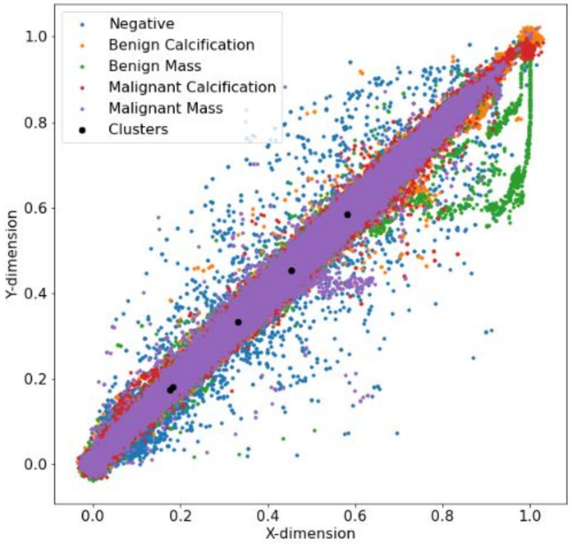
Results - Multi-Class with K-means

AUC_ROC: 0.4908

Confusion Matrix score: -2476

Accuracy: 18.5333



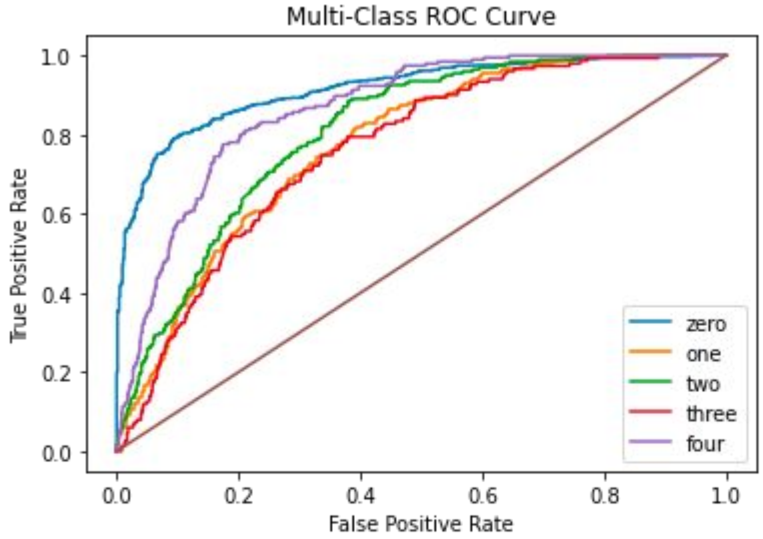


Results - Multi-Class with CNN

AUC_ROC: 0.8712

Confusion Matrix score: 98

Accuracy: 60.8



Conclusion

- Machine learning projects should utilize data augmentation to increase their dataset and generalizability
- K-means algorithm is not a suitable supervised learning classifier
- RandomForest is a reliable model