# Facial Keypoints Recognition

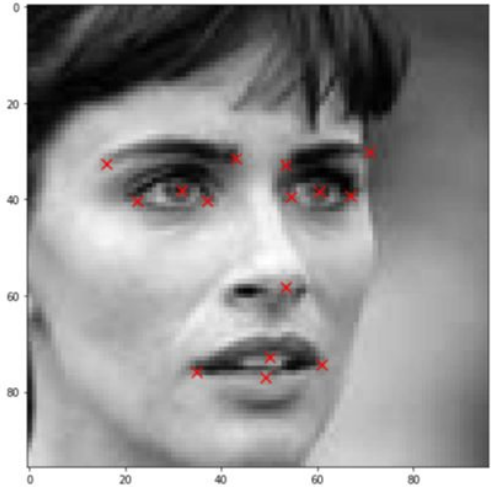
**Final Project W207 | Dr. Yacov Solomon | Monday 18:30 Pacific**

**Andrew Webb, Laura Chutny, Suzy Choi, and Yue Hu**

**5 August 2019**

# Goal

- Input: X: Facial Image
- Output: Prediction of the coordinates of 15 specific facial keypoints in a previously unseen image.
- Score: The lowest root mean squared error (RMSE) classifying facial keypoint locations in digital grayscale images

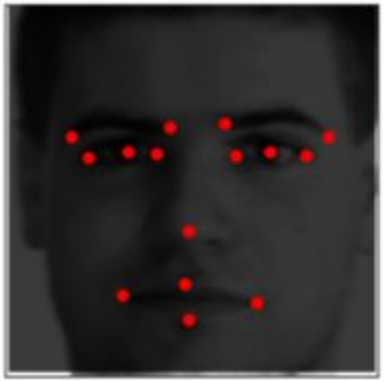


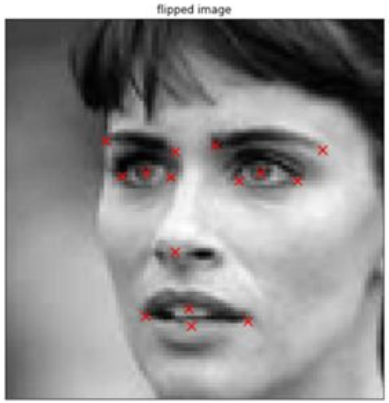
# Data Exploration

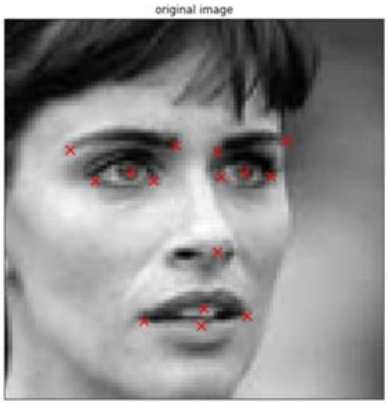
- Training data has 7049 images, some fully and some partially labeled.
  - 1842 training data and 298 dev data with complete labels
- The x and y position of 15 facial keypoints in an image result in 30 columns of label names and 30 points with pixel locations to correctly classify for each test image.
- Problems with Images
  - Images that are not of people
  - Images that are not centered
  - Images that only show 1 side of a person
  - Images that only have partial labels

# Feature Engineering

- Blurring the image
- Accentuating the Image
- Flipping image to artificially create more training data



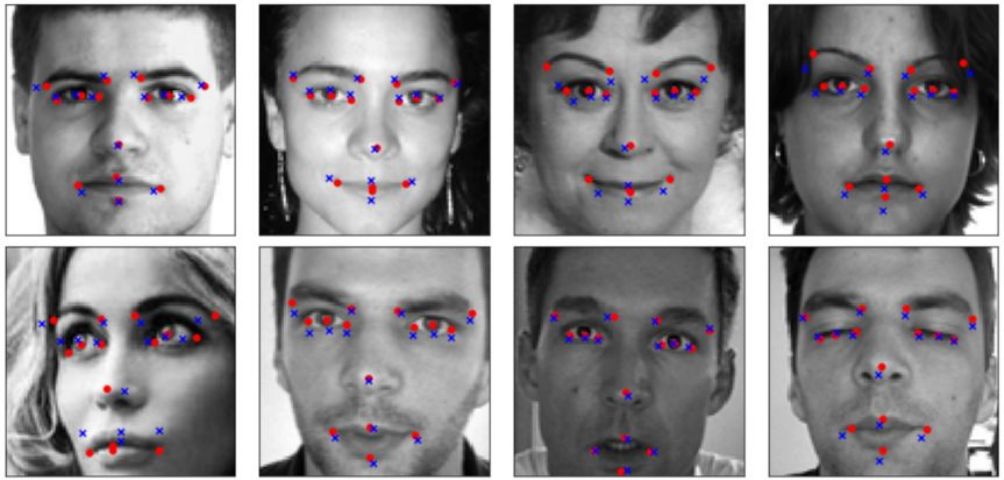


# Model1: K-Nearest Neighbors

- Model: K-Nearest Neighbors
- Euclidean Distance
- Optimum k=5
- Cost and Objective: Non-parametric, lazy learner model
- K and distance are hyperparameters
- Scoring: RMSE

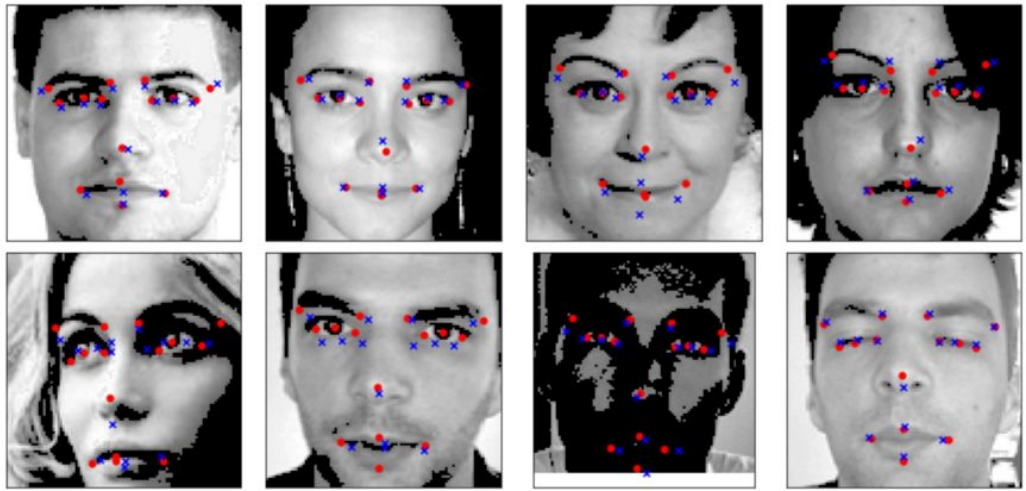
→ Original RMSE: 0.05225
→ Accentuated RMSE: 0.05594
→ Blurred RMSE: 0.05225
→ Flipped RMSE: 0.05201



# Model 2: Linear Regression

- Model: Linear Regression
- Cost Function: Mean Square Error
- Objective: Minimize uniform average MSE
- Methodology: Gradient Descent
- Scoring: RMSE

→ Original RMSE: 0.05693
→ Accentuated RMSE: 0.05142
→ Blurred RMSE: 0.06409
→ Flipped RMSE: 0.06705

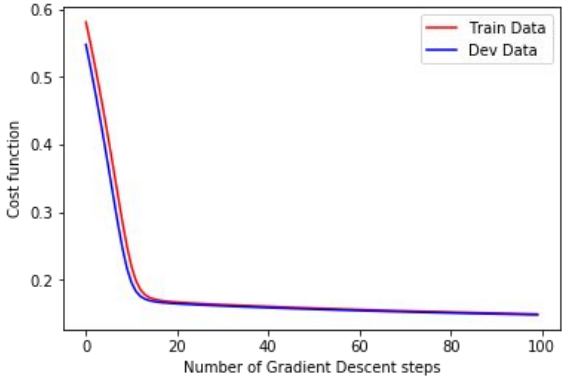


# Model 3: Single-Layer Neural Network

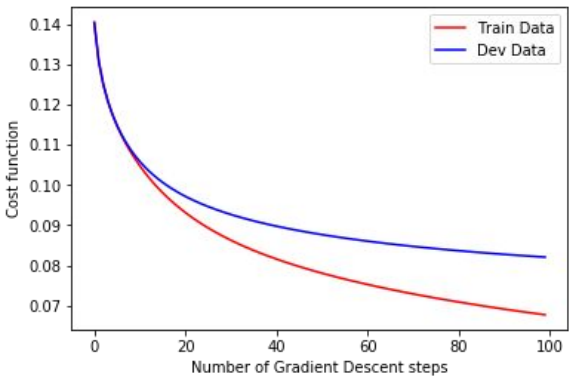
- Model: Single layer neural network with **tanh activation**/ linear regression
- Cost Function: RMSE
- Objective: Minimize RMSE
- Methodology: batch gradient descent/ stochastic gradient descent / **mini-batch gradient descent (batch size =10)**
- Learning Rate: 0.01 / **0.001**

→ RMSE: 0.082

**Batch Gradient Descent**



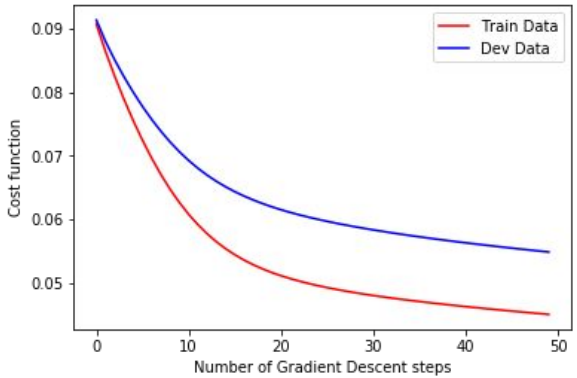

**Mini-batch Gradient Descent**

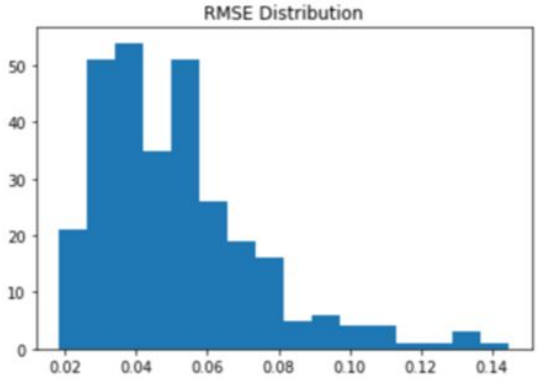


# Model 4a: Multi-Layer Neural Network

- Model: Multi-layer neural network
  - **One hidden layer**/ three hidden layers
  - **600 hidden nodes**/ 800 hidden nodes
  - Activation Function: **Tanh**/ Rectifier activation function in hidden layer
- Cost Function: RMSE
- Objective: Minimize RMSE
- Methodology: mini-batch gradient descent
- Learning Rate: 0.001

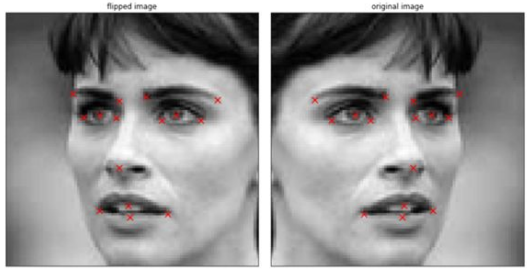
→ RMSE: 0.05



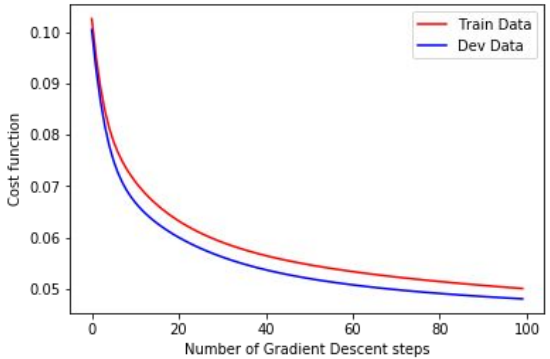



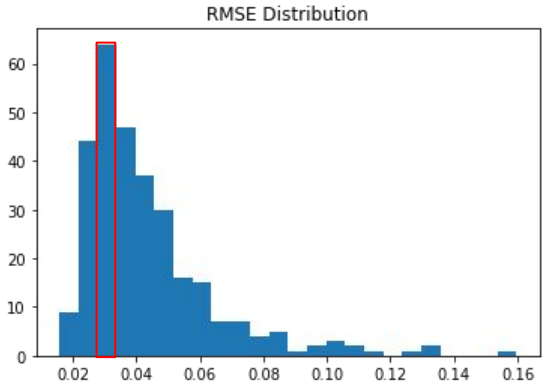
# Model 4b: Multi-Layer Neural Network

- Train Data: Combination of Original Image with **Flipped** Image with **Accentuating**
- Model: Multi-layer neural network with tanh activation
  - **<u>One hidden layer</u>**, **<u>600 hidden nodes</u>**
  - Activation Function: **<u>Tanh</u>**
  - Cost Function: RMSE
- Objective: Minimize RMSE




→ RMSE: 0.048



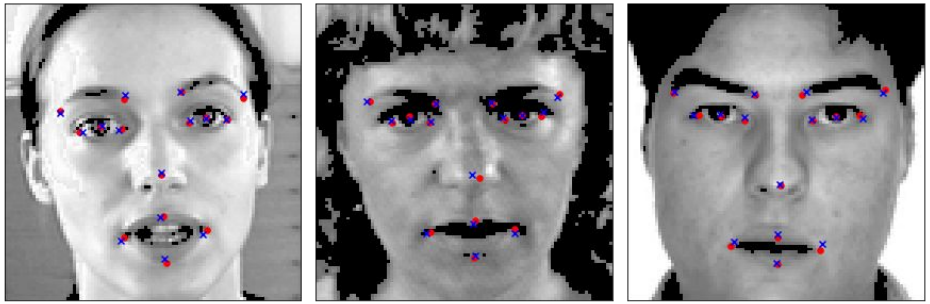
# RMSE Comparison

| | Complete Labels | Accentuate Images | Blur Images | Add Flipped Images |
|---|---|---|---|---|
| KNN | 0.052 | 0.056 | 0.052 | 0.052 |
| Linear Regression | 0.057 | 0.051 | 0.064 | 0.067 |
| Single-Layer NN | 0.082 | | | |
| Multi-Layer NN | 0.050 | 0.050 | 0.051 | 0.048 |

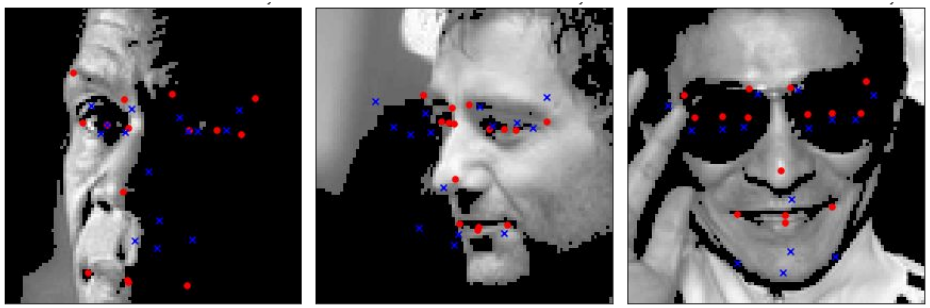
Combining Multi-Layer NN: blurred images with larger training set (flipped images) = 0.0477 RMSE (BEST)

# Future Plan for Further Improvement

Images with Lowest RMSE



Images with Highest RMSE



Feature engineering
Images with low prediction accuracy:

- Image showing face from side
- People wearing sunglasses
- Images with low resolution
- Images with faces not centered

Modeling
Convolutional Neural Net - next step
Need speed improvement to run effectively