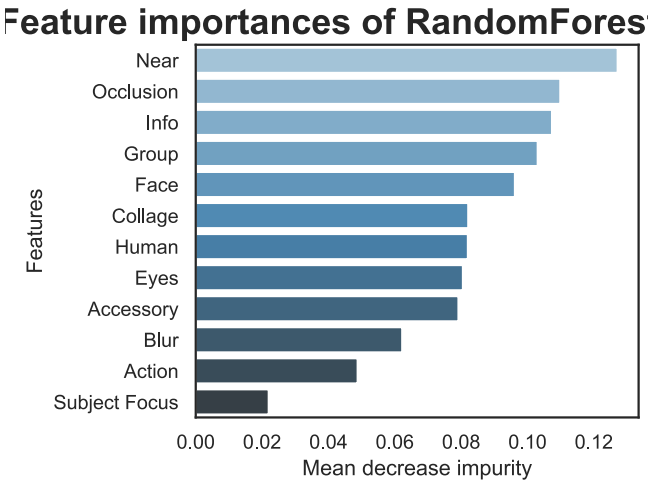
Random forest regression on the metadata

1. Used *RandomForestRegressor* library in Python.
2. Tuned three parameters: *bootstrap*, *max_features* and *n_estimators.*
3. The best values for these three parameters are: *True, log2, 500.*
4. Got a 20.64 RMSE.
5. Feature importance plot.
   Near, Occlusion, Info, Group, Face are the top 5 important features.

Near: Single pet taking up significant portion of photo.
Occlusion: Specific undesirable objects blocking part of the pet.
Info: Custom-added text or labels.
Group: More than 1 pet in the photo.
Face: Decently clear face, facing front or near-front.



Basic CNN on the image data

1. Used *tensorflow, keras* library in Python.
2. Built the CNN model:

```
inputs=keras.Input(shape=(image_width,image_height,3))
x=inputs
x=keras.layers.Conv2D(filters=16,kernel_size=3,strides=2,padding='same',activation='relu')(x)
x=keras.layers.MaxPool2D(pool_size=(2, 2))(x)
x=keras.layers.Conv2D(filters=32,kernel_size=3,strides=2,padding='same',activation='relu')(x)
x=keras.layers.MaxPool2D(pool_size=(2, 2))(x)
x=keras.layers.Conv2D(filters=64,kernel_size=3,strides=2,padding='same',activation='relu')(x)
x=keras.layers.MaxPool2D(pool_size=(2, 2))(x)
x=keras.layers.Conv2D(filters=128,kernel_size=3,strides=2,padding='same',activation='relu')(x)
x=keras.layers.Flatten()(x)
x=keras.layers.Dense(128, activation = "relu")(x)
x=keras.layers.Dropout(0.2)(x)
output = tf.keras.layers.Dense(1)(x)
model = tf.keras.Model(inputs=inputs, outputs=output)
```

3. Fit on train data and validated on test data.
4. RMSE train/validation by Epoch plot.
   The validation RMSE stabilized at 20.68.