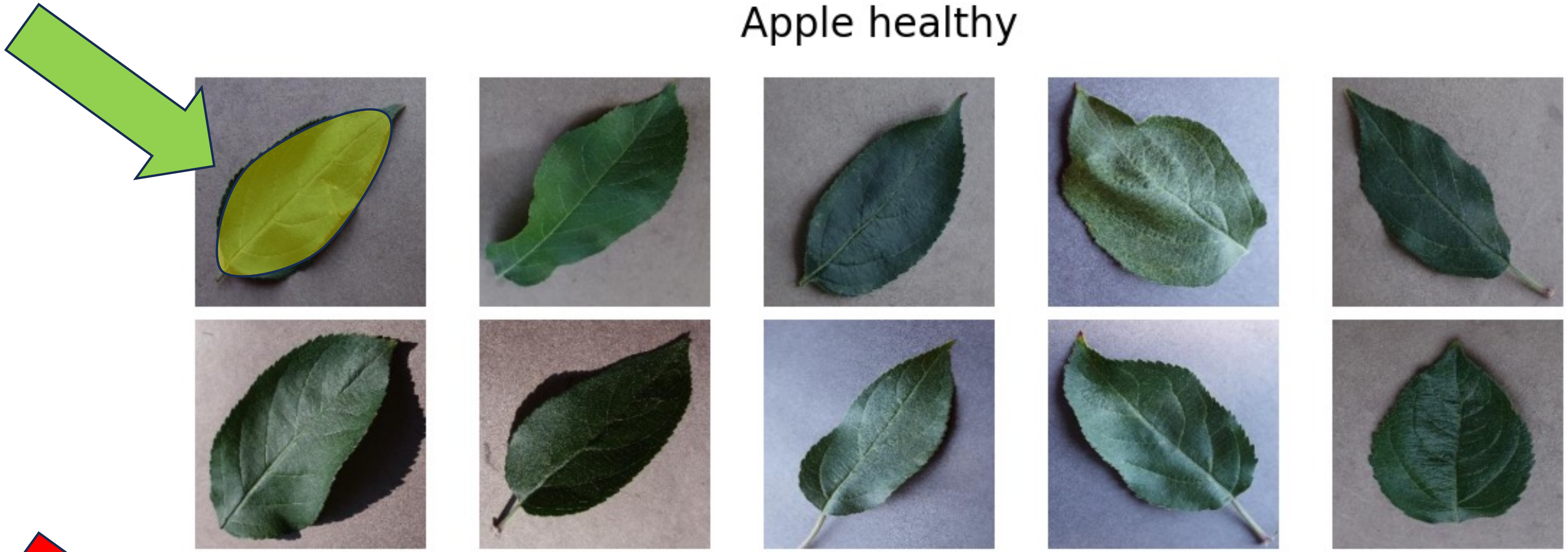
# Getting Insights into Images and their Metadata

Lab ML for Data Science: Part III

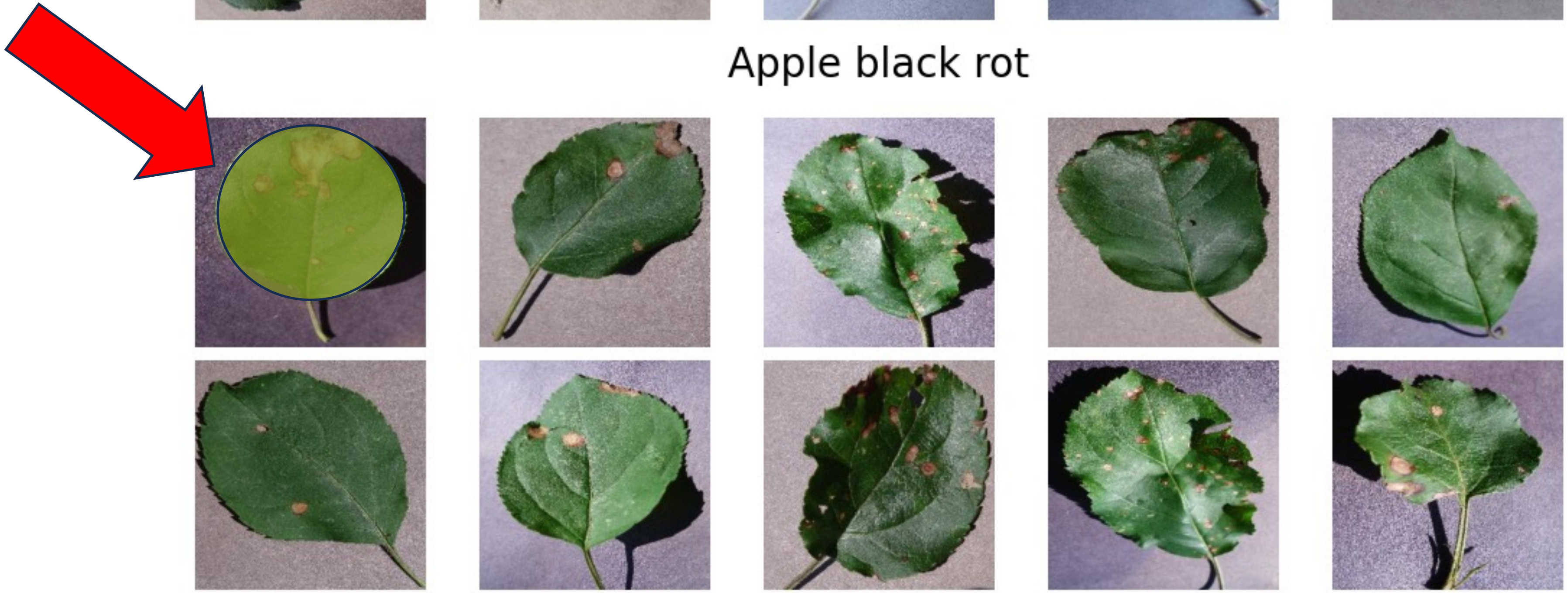
# Goal for the Project

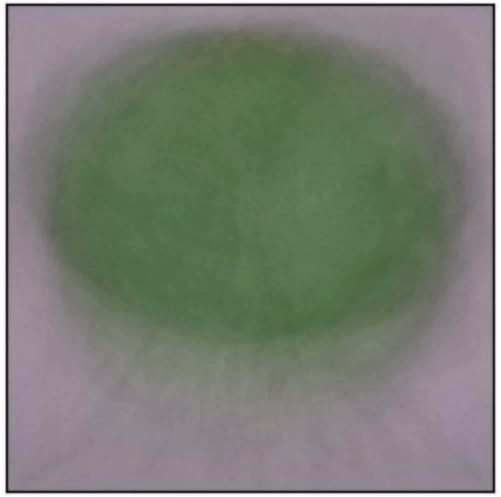
1. Given an image of an Apple Leaf predict class (healthy/sick)
2. Derive explainations for predictions
3. Discuss results

## Apple healthy

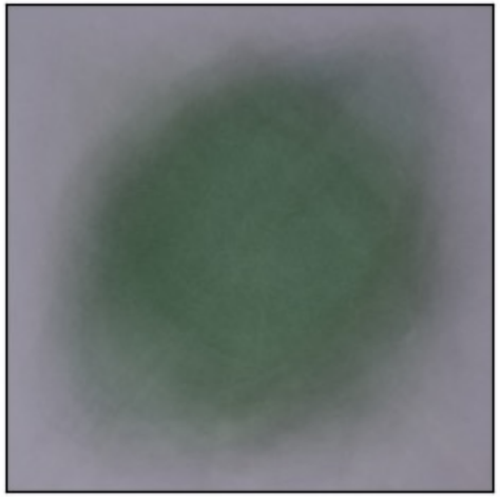


## Apple black rot



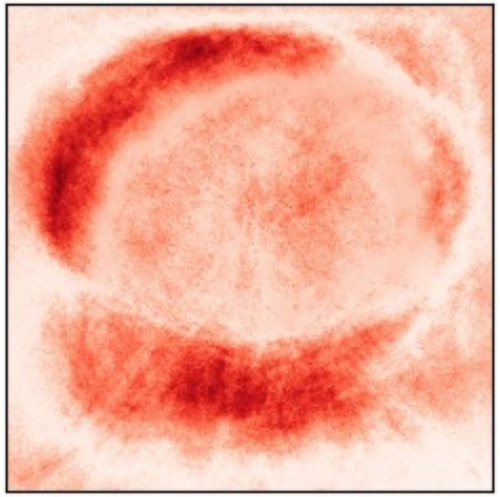


Mean apple black rot

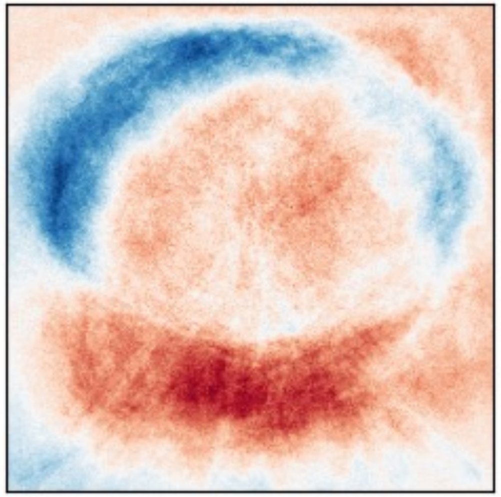


Mean apple healthy

Subtracting

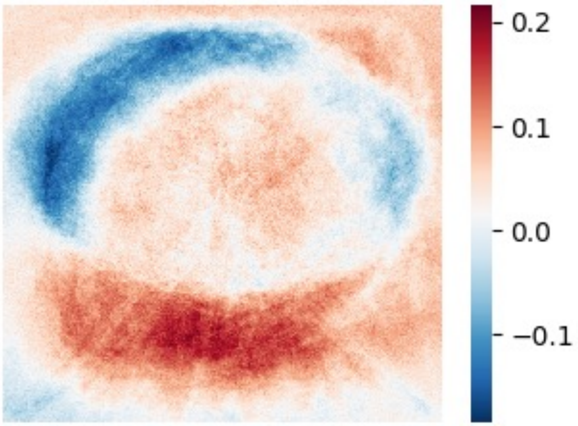


Difference of the means
sum over channels

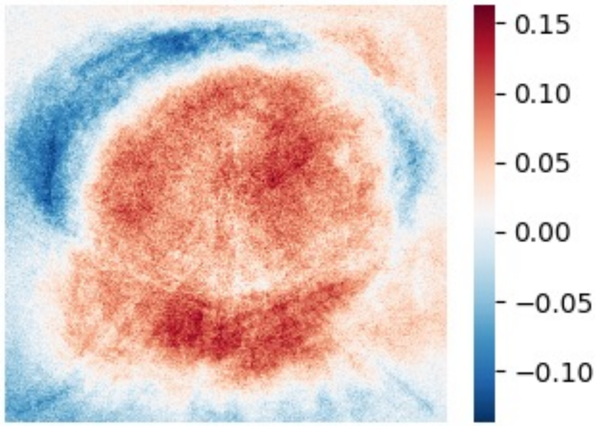


Difference of the means
norm over channels

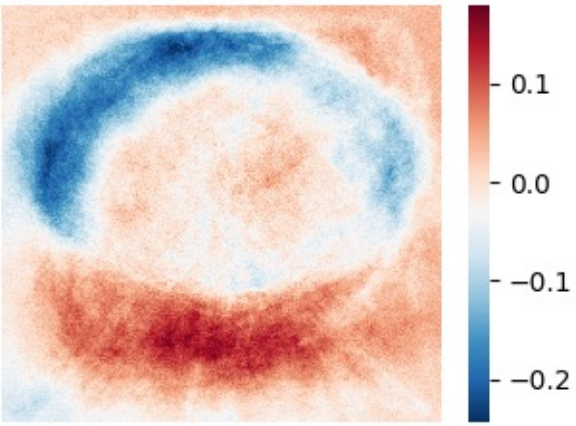
Mean differences per channel



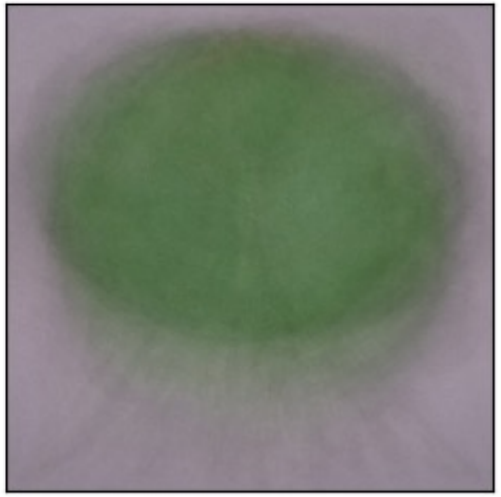

Channel 0: Red



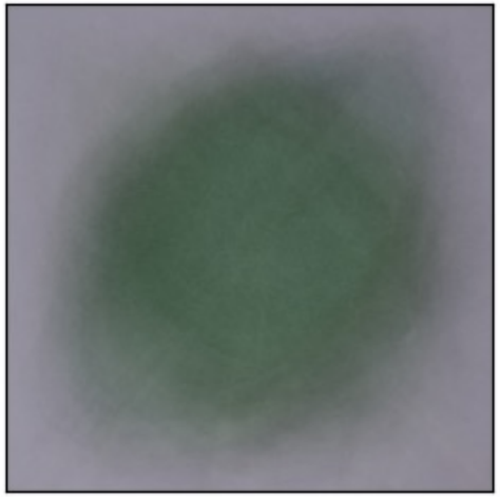

Channel 1: Green




Channel 2: Blue

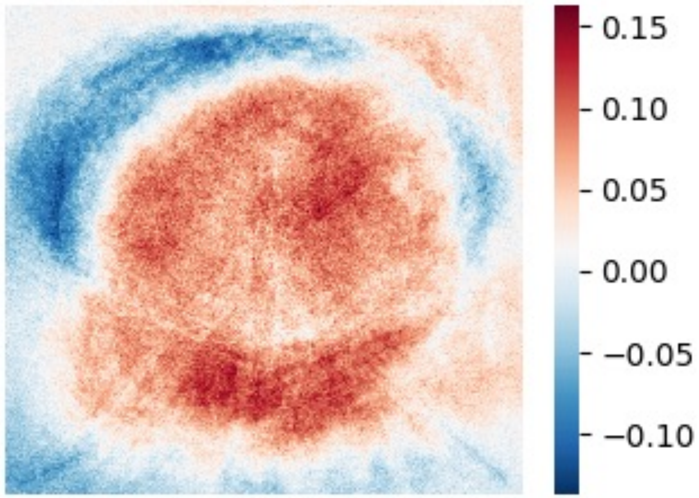


Mean apple black rot



Mean apple healthy

Subtracting




Channel 1: Green

Observation:

- Shape (& slightly Position) vary on average for the classes
- Healthy: more oval + „pointy“ tip
- Black Rot: more round + „flat“ tip

Possible causes:

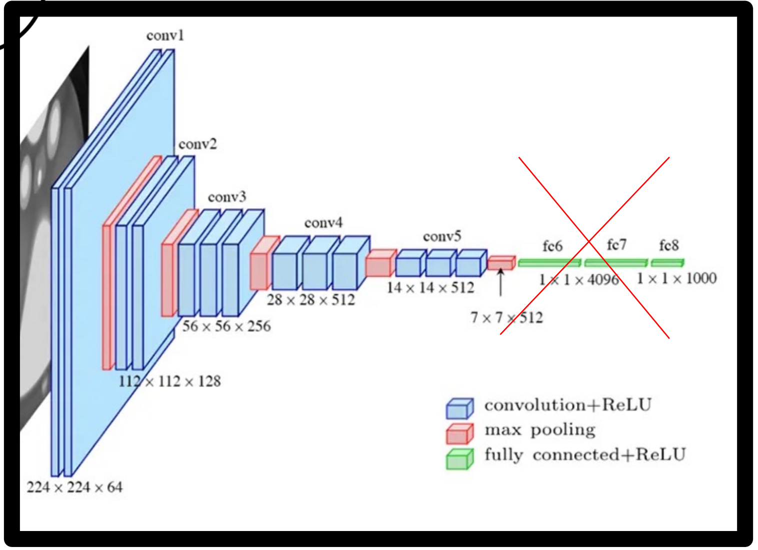
- Shape: Sampling bias
- Tip: Disease leads to detoriated tip

# Pipeline for making Predictions

1 Input

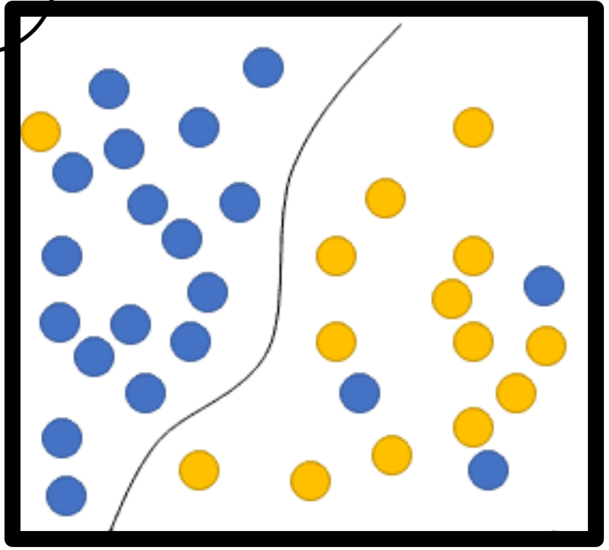
- As scaled Tensor (values in [0,1])
- Eventually Standardized (on train & VGG-16)

2 Feature Extraction




- Pretrained VGG-16
- ~15M Paramaters
- Discarding classification head
- Add Flatten layer

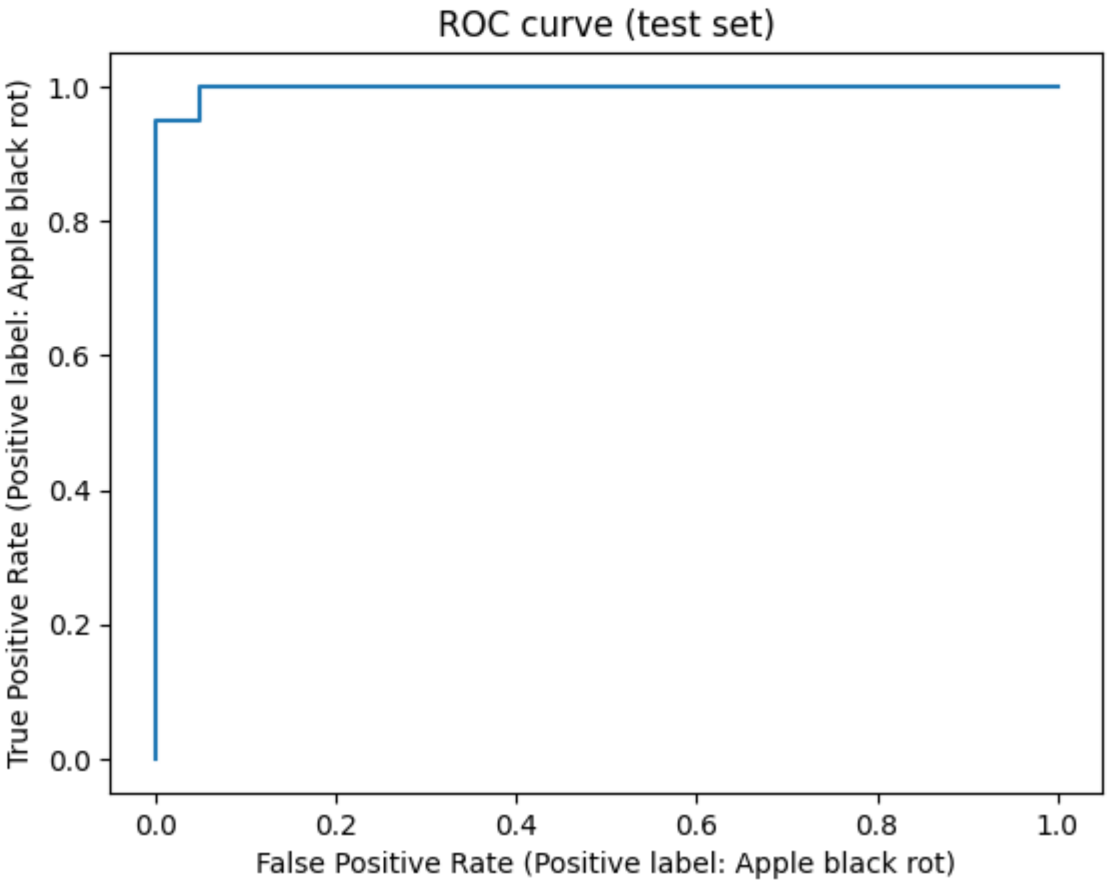
3 Classification



- Difference of mean discriminant

# Technical Details

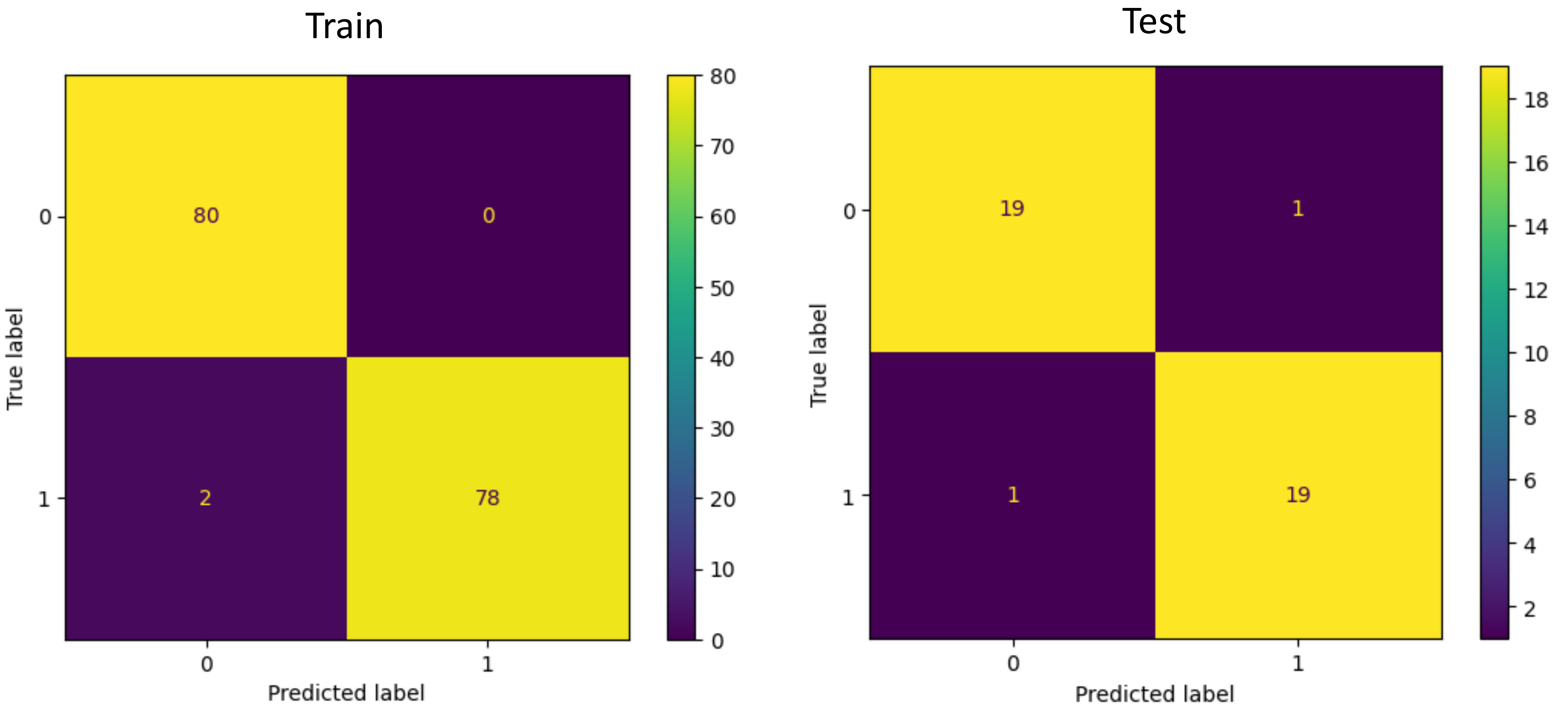
# Results




- Stratified Train/Test-Split of 80%/20% on unstandardized data
- 80 instances of each class in train, 20 of each in test
- AUC Train: ~0.998
- AUC Test: ~0.997

# Results

- Optimizing threshold for maximum accuracy leads reaching a test accuracy of 95%



# Understanding the Class-Prediction Pixel-Wise

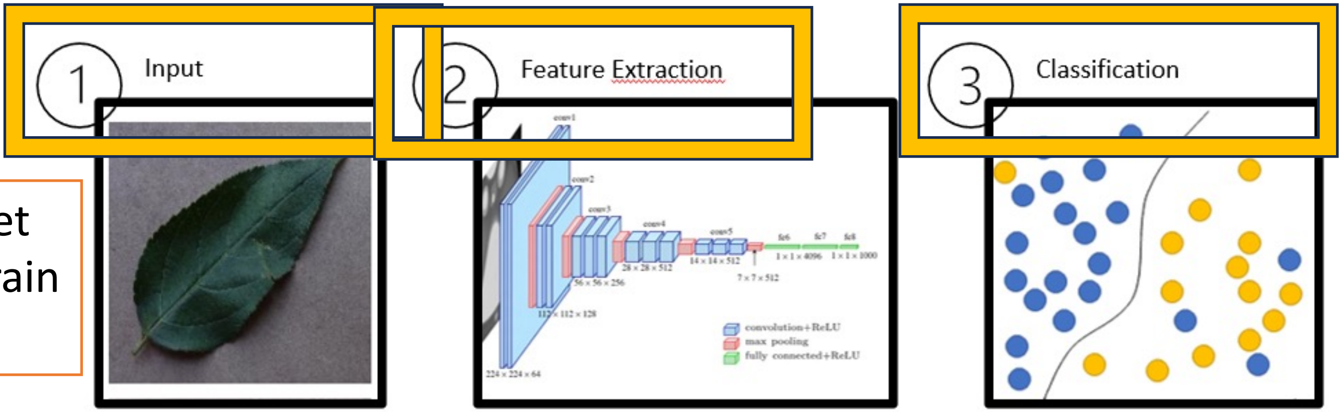
- Derive exact features which lead to predictions
- Enable to validate predictions exluding predictions based on artefacts (keyword: Clever Hans Effect)
- Gain knowledge about the relationship between features and class

- Most intuitive Basis: $S_i = \left\| \frac{\partial g}{\partial x_i} \right\|^2$ → getting Relevance for each pixel

# Understanding the Class-Prediction Pixel-Wise

- Most intuitive Basis: $S_i = \left\| \frac{\partial g}{\partial x_i} \right\|^2$ → getting Relevance for each pixel
- This approach might get noisy results
- To mitigite that problem we fiddle around in the pipeline, mainly:

1. Standardize on Training Set
2. Standardize on VGG-16 Train
3. Add Noise




- As scaled Tensor (values in [0,1])

- Pretrained VGG-16
- Trained on ImageNet
- Discarding classification head
- Add Flatten layer

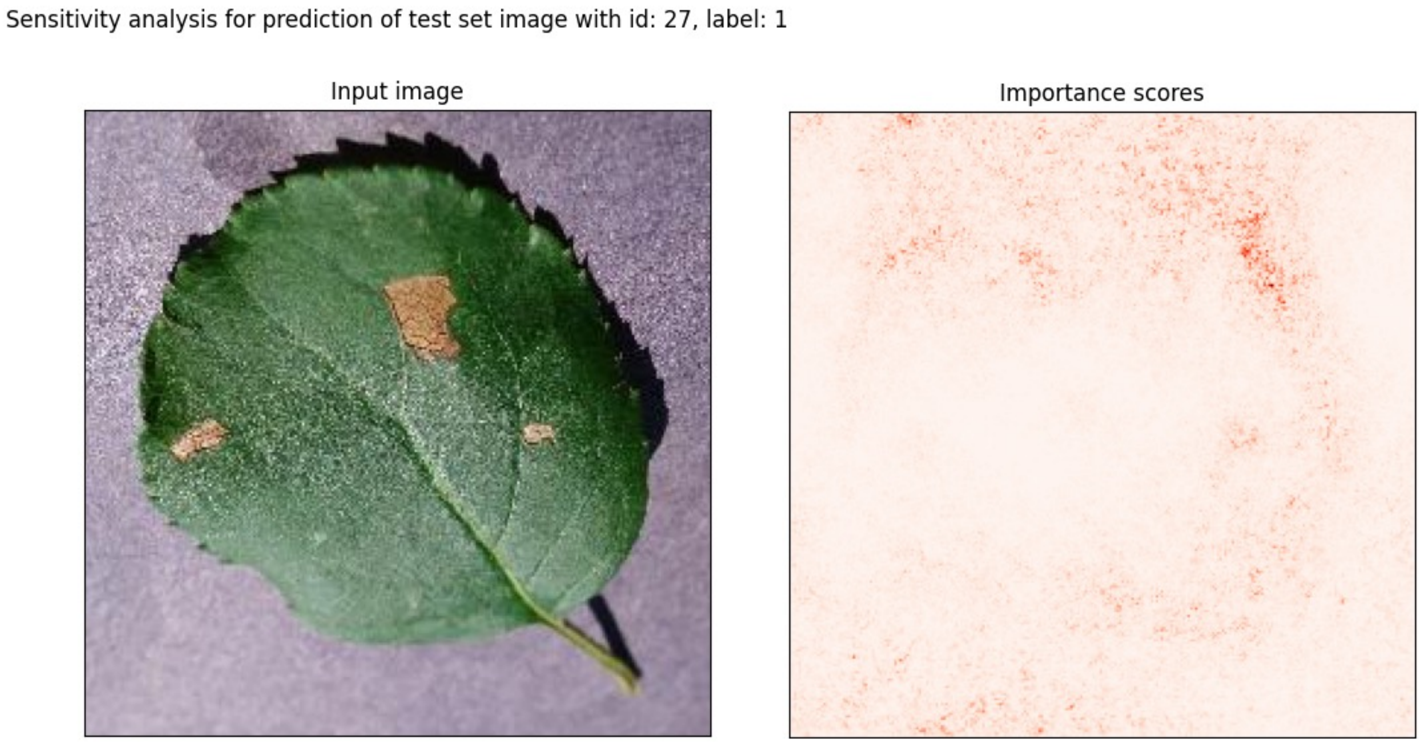
- Difference of mean discriminant
- I.e. projecting extracted features of input on difference of means for each class

1. Bias Gradient of Convolutional Layers

$$z_k = \left(\sum_j a_j w^{\uparrow}_{jk} + b^{\uparrow}_k\right) \cdot \left[\frac{\sum_j a_j w_{jk} + b_k}{\sum_j a_j w^{\uparrow}_{jk} + b^{\uparrow}_k}\right]_{\text{cst.}}$$

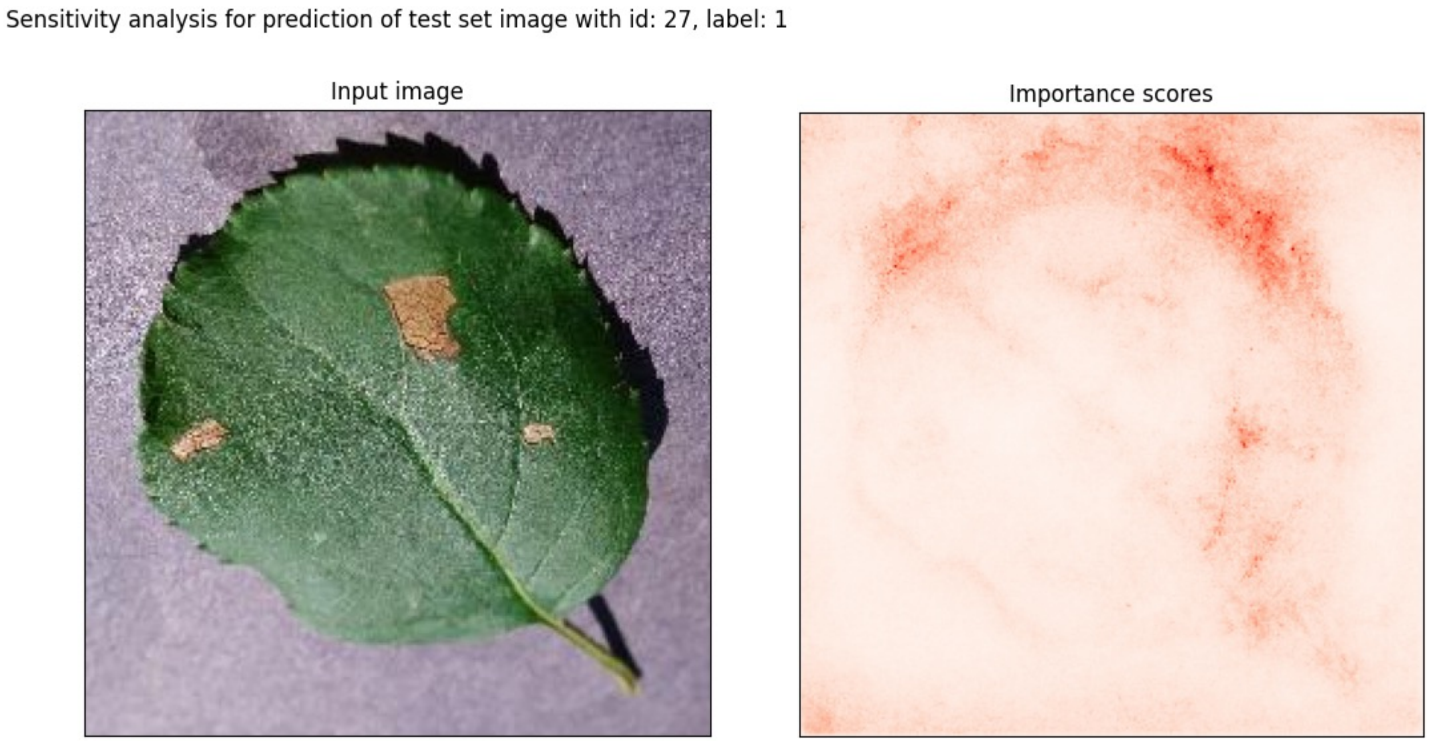
$$w^{\uparrow}_{jk} = w_{jk} + 0.25 \max(0, w_{jk})$$
$$b^{\uparrow}_k = b_k + 0.25 \max(0, b_k).$$

# Sensitivity Analysis - Simple Approach




- Unstandardized data
- Delivers noisy results
- Arguabely detect 1 spot

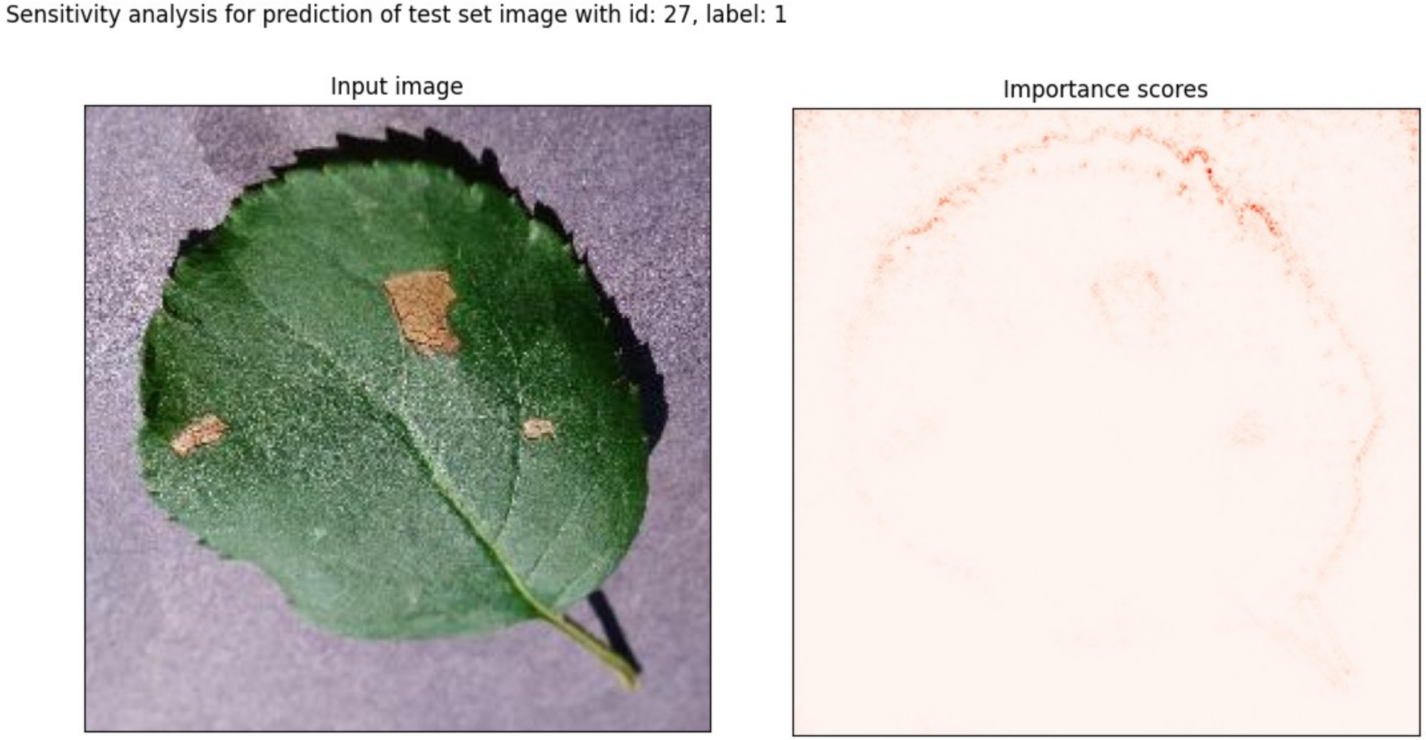
# Sensitivity Analysis – SmoothGrad




- Gaussian Kernel with

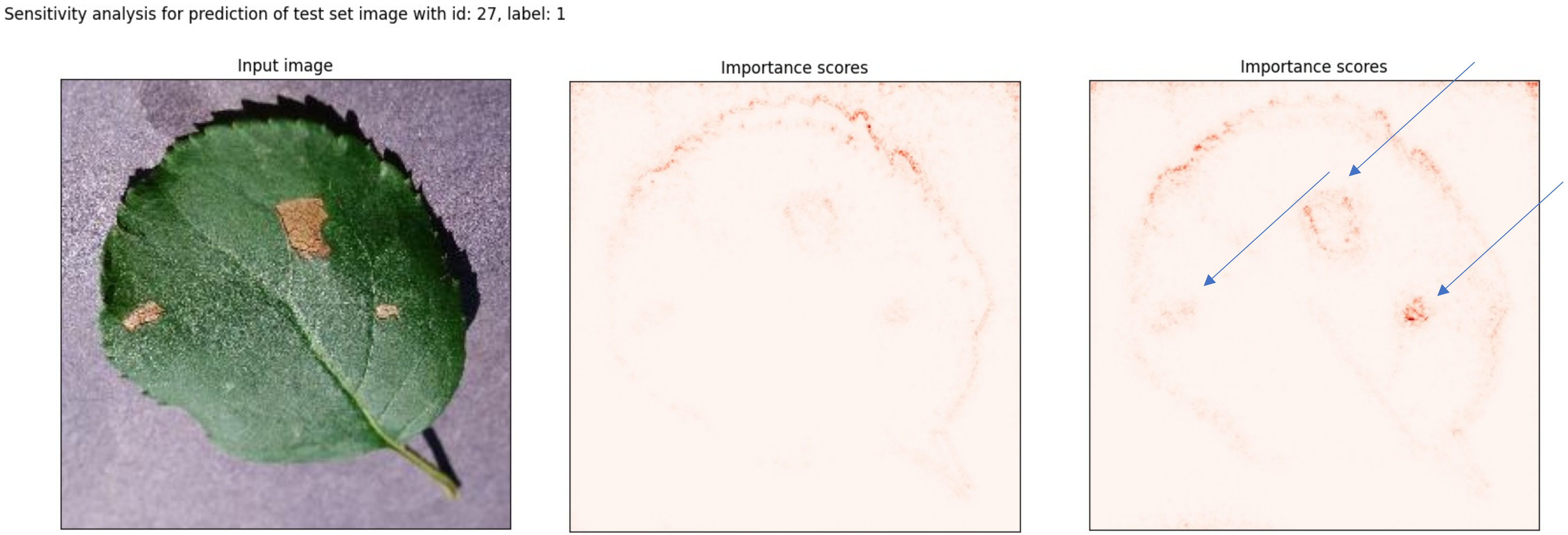
  sd = 0.1 and N=20
- Now contours detected as well as black-rot spot (?)
- At cost of slightly more noise

# Sensitivity Analysis – Biased Layer Approach



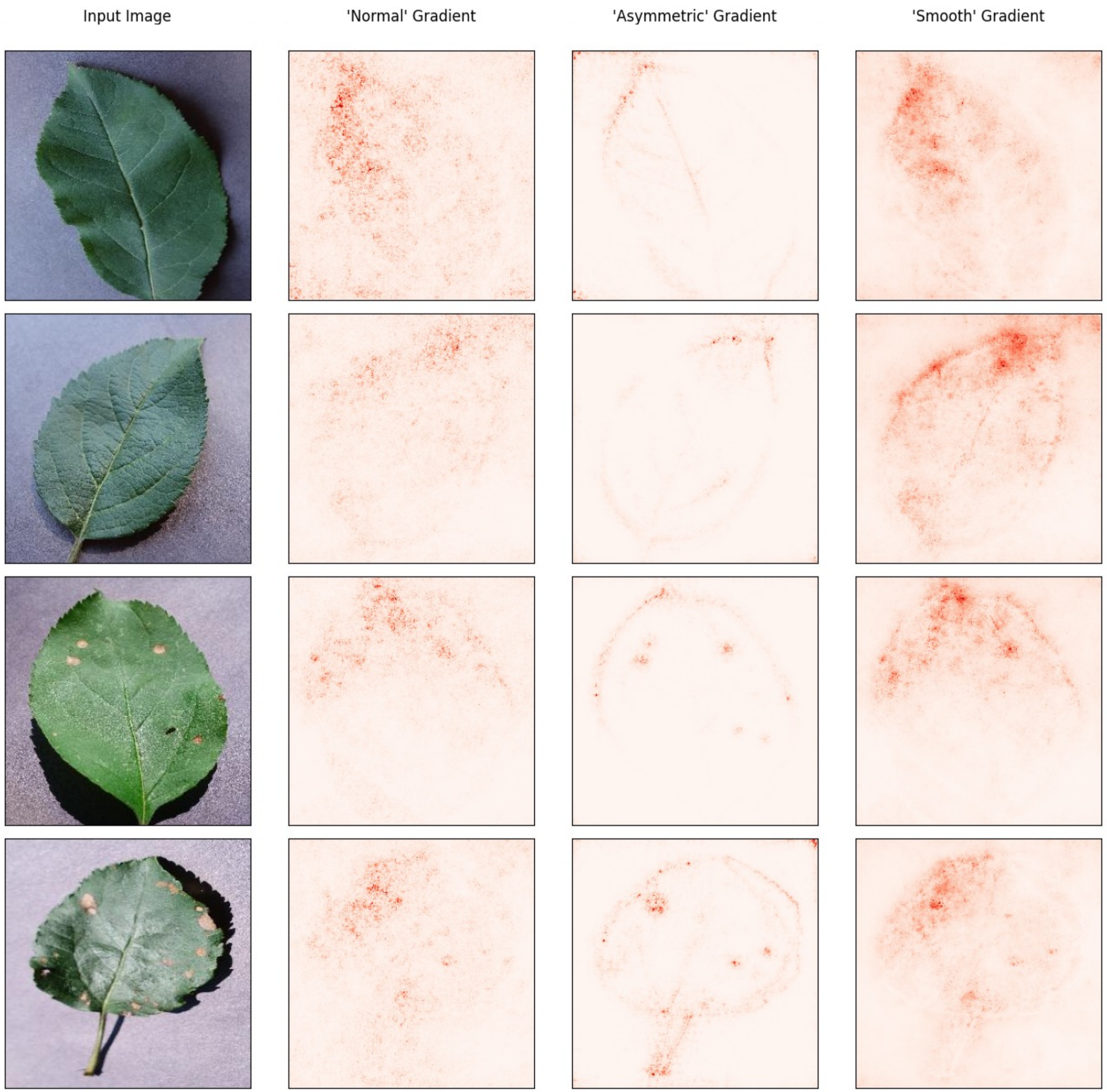

- Way less noiser explaination
- „More global“ explaination
- Excitory > Inhibitory effects
- Less important pixels diminished
- Anomalous spots all detected
- → Try to Improve with standardized data

# Sensitivity Analysis – Improve with Standardizing

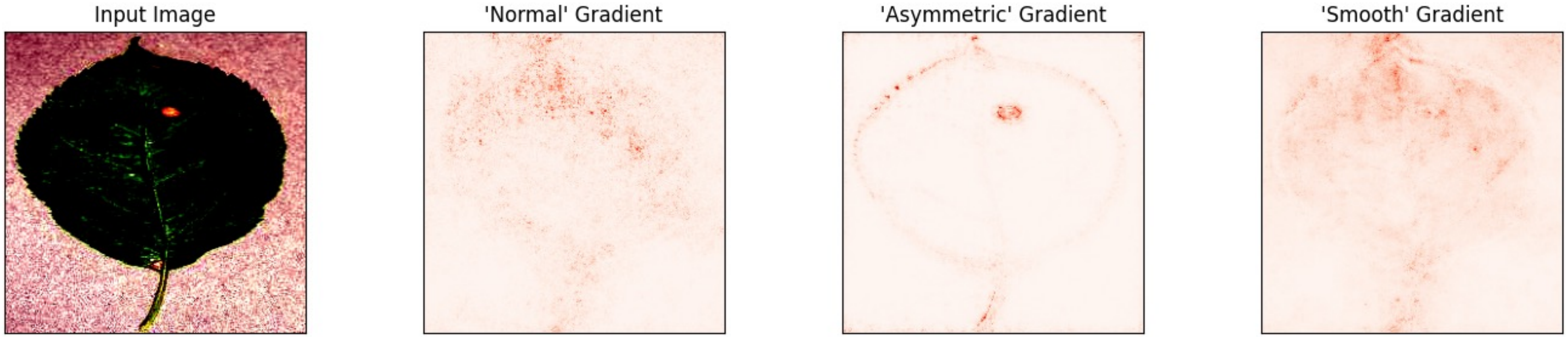



„Raw“/only scaled

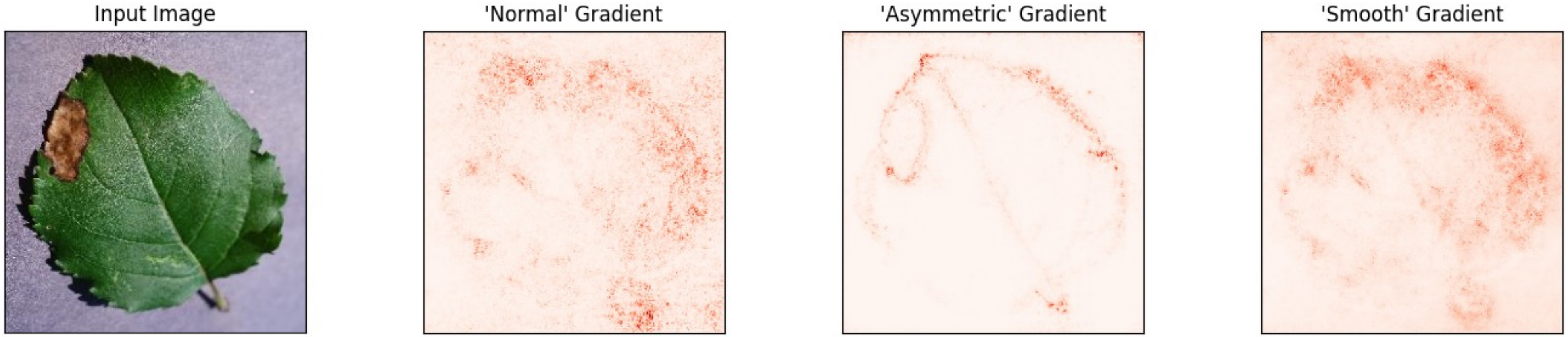
Standardized on Train Set


Input Image
'Normal' Gradient
'Asymmetric' Gradient
'Smooth' Gradient

# Miscellaneous




- Misclassified
- Shape?




- Sanity Check with sheet

# Discussion of the Results

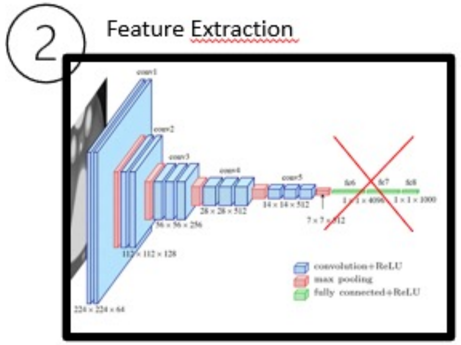
# Discussion:
# Insufficiently good pretrained neural network

Description:

- Model in Pretext Task not trained well enough → features not sufficiently learned

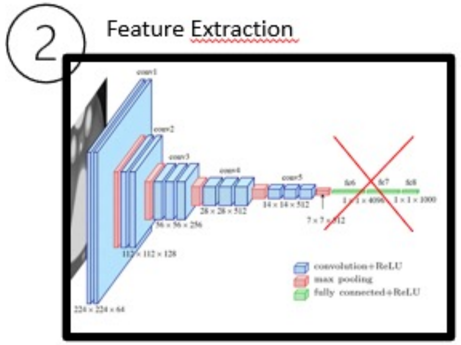
Possible Solution:

- Check model performance on Prext Task
- Check if similar downstream task exists
- Fine-Tune model parameters on task at hand
- Experiment with other pretrained NNs



# Discussion:
# Improper Method for extracting relevant features

Description:
- Each layer of pretrained Model captures different aspects of image
- Last layer may be inaccurate choice for our task
- Contours/dark areas dominant in Sensitivity analysis
- Data domain too different from leafes




Possible Solution:
- Try cutting out more layers to extract appropriate features
- Fine-tune on dataset at hand
- Experiment with other pretrained NNs

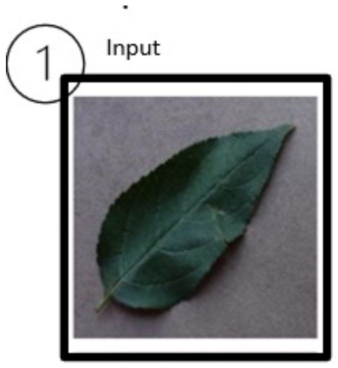
# Discussion:
# Problems with data quality

Description:

- Potential bias in data sampling (as seen with the means)
- Image resolution/sharpness to low (vague features unable to capture detailed structures)
- Shadows possibely introduce noise hardening the detection of relevant features

Possible Solution:

- Sample more data
- Manually sample images to balance types of leaf shapes or ignore shadows
- Increase sharpness of images with designated ML models
- Remove shadows



# Discussion:
# Flawed Domain Knowledge of Human

Description:

- Disease might affect plant in a way that is unkown by humans
- Detected factors for the disease might not be perceivable by humans

Possible Solution:

- Consult experts with domain knowledge and double check results
- Investigate possible newly detected symptoms