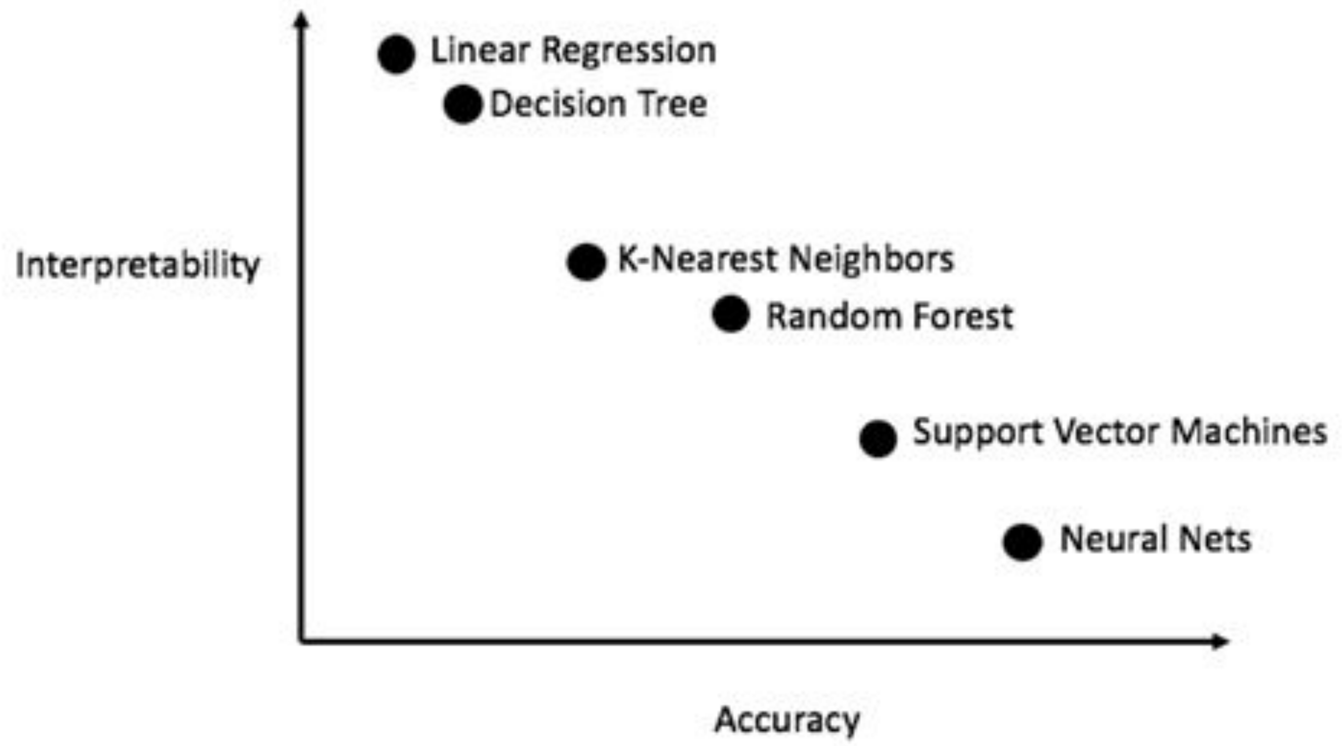
Exploring Explainability Methods using Trashnet Model

Erald Keshi

Explainability in AI

- Lack of understanding of model underlying behaviour stops ML/AI adoption in sensitive industries.
- Build trust in the model before deploying it
- Really understand what complex models have learned
- Present explanations in intuitive and simple way

Interpretability vs Accuracy



TrashNet Problem

1. Image Classification problem
2. Classify images of trash into 6 categories : paper, cardboard, trash,glass, metal, plastic.
3. Around 2200 images

Model used

1. Deep Neural Network
2. Model was reused from https://github.com/vasantvohra/TrashNet
3. Model layers are not important since model will be treated as black box

Explainability Methods Explored

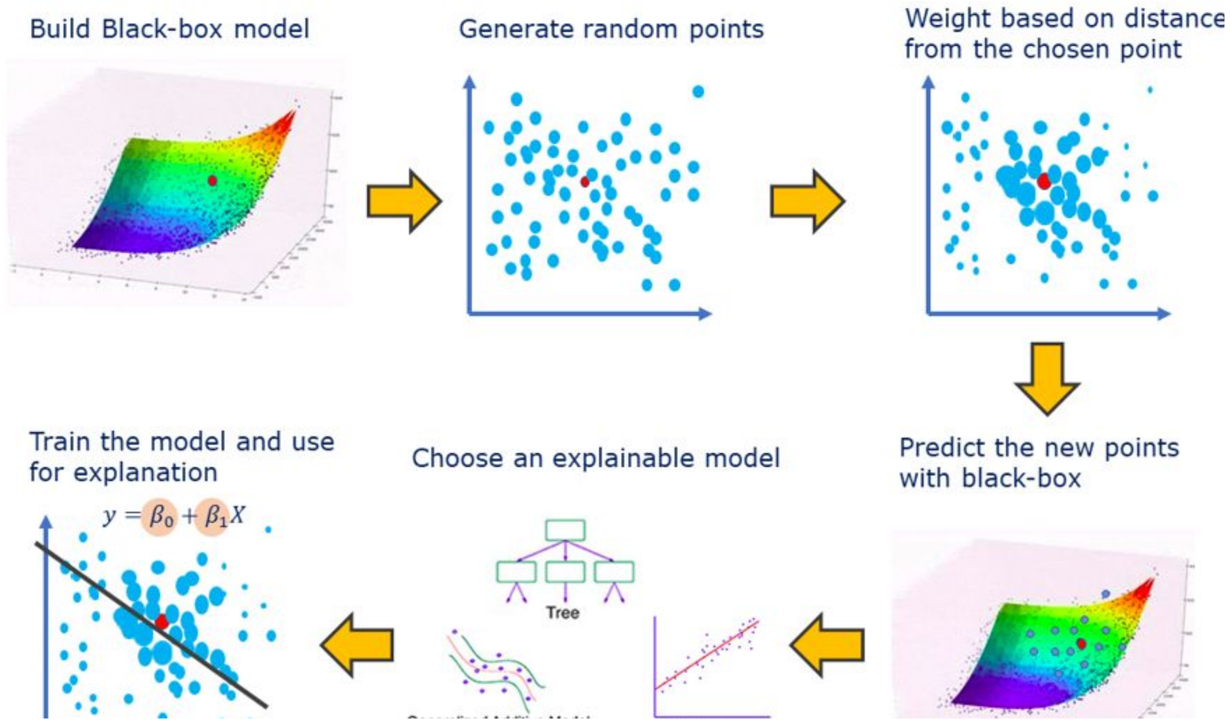
1. LIME : Local Interpretable Model-agnostic Explanations
2. SHAP: Shapley Values
3. Occlusion Sensitivity Mapping

LIME

LIME - Local Interpretable Model-agnostic Explanations

Model agnostic, which means that LIME is model-independent and is able to explain any black-box classifier.
Interpretable, which means that LIME provides you a solution to understand why your model behaves the way it does.
Local, which means that LIME tries to find the explanation of your black-box model by approximating the local linear behavior of your model.

LIME - Local Interpretable Model-agnostic Explanations



https://towardsdatascience.com/lime-explain-machine-learning-predictions-af8f18189bfe

SHAP

Shap - Key characteristics

1. Based on Shapley values in Game Theory
2. Model Agnostic
3. Local explanations
4. Operates similarly to LIME , both tweak input data and observe differences in results.
5. Expensive to brute force, needs to be approximated.

How SHAP works

Parallelism between Game Theory and AI

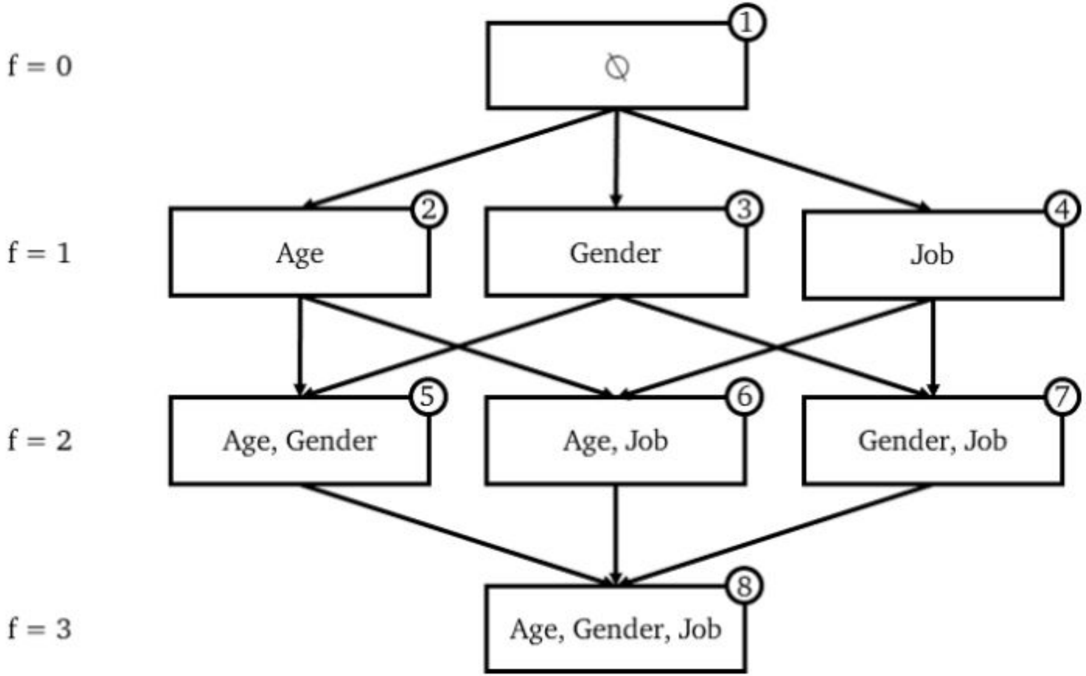
Game -> Predicting outcome of the model

Players -> Features

SHAP quantifies the contribution that each feature brings to the prediction made by the model.

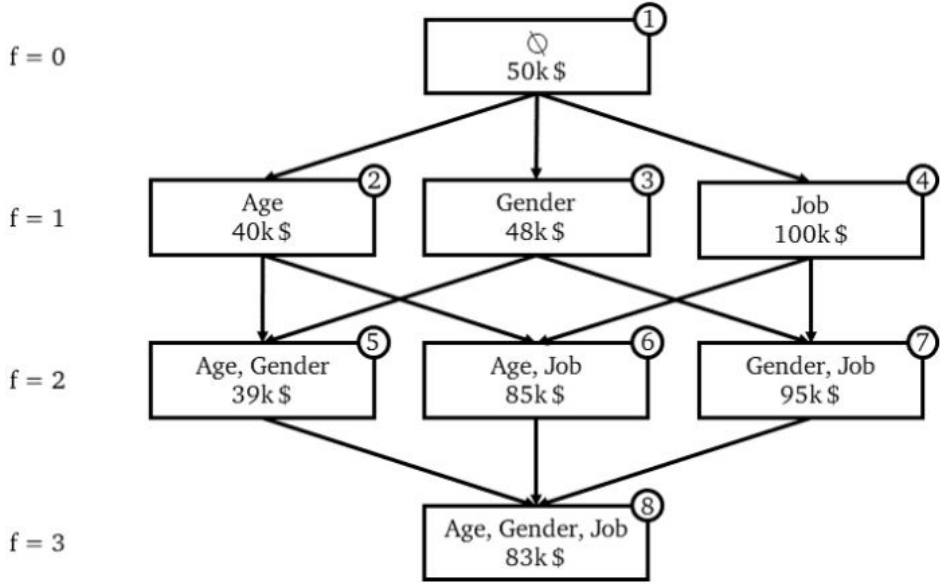
https://towardsdatascience.com/shap-explained-the-way-i-wish-someone-explained-it-to-me-ab81cc69ef30

How SHAP works



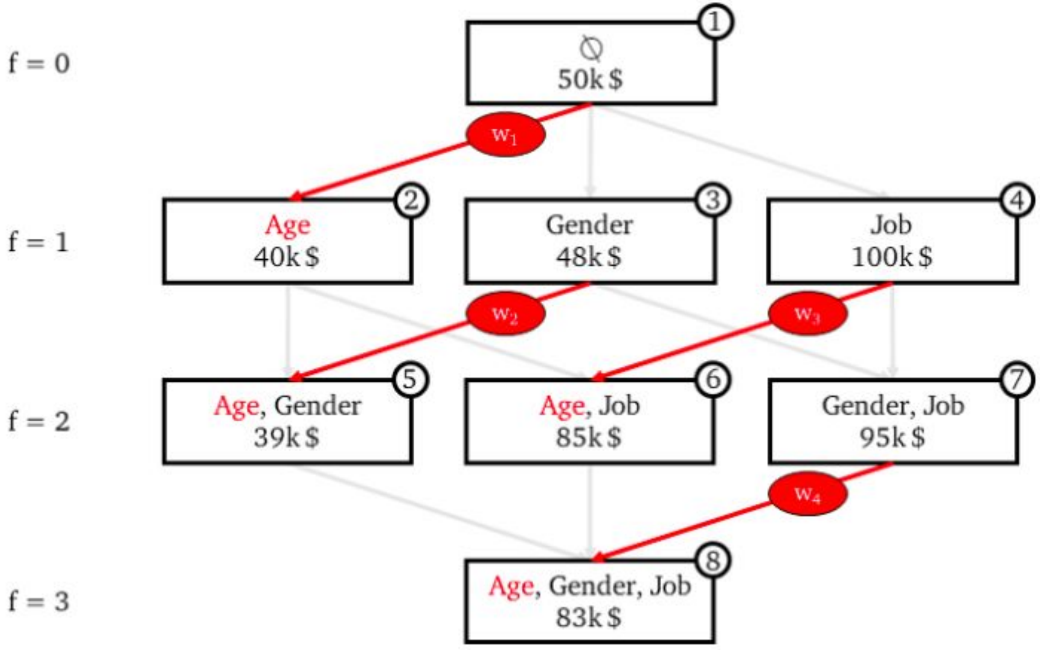
Power set of features

SHAP requires to train a distinct predictive model for each distinct coalition in the power set, meaning 2 ^ *F* models. These models are completely equivalent to each other for what concerns their hyperparameters and their training data (which is the full dataset). The only thing that changes is the set of features included in the model.



Predictions made by different models for x_0. In each node, the first row reports the coalition of features included in the model, the second row reports the income predicted for x_0 by that model.

Weighted average of marginal contributions of a feature = SHAP value for that feature

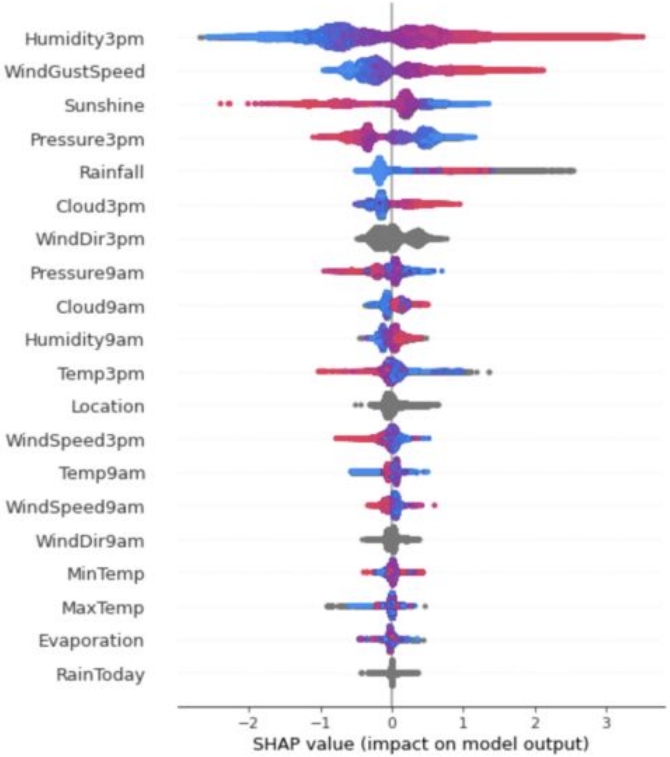


Marginal contributions of Age

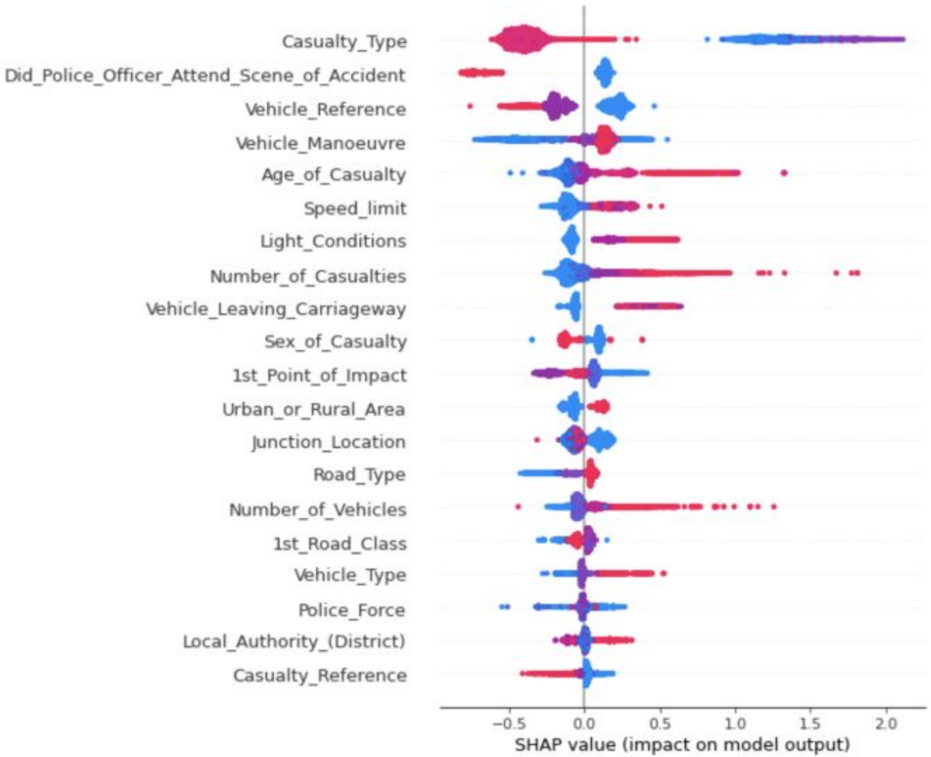
- SHAP_Age(x_0) = -11.33k $
- SHAP_Gender(x_0) = -2.33k $
- SHAP_Job(x_0) = +46.66k $

How visualizations look with SHAP

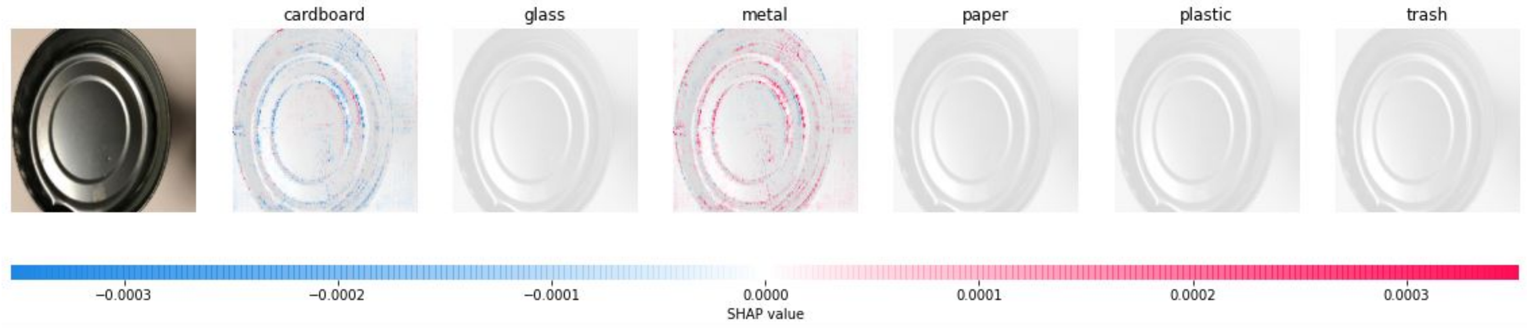
Will it rain tomorrow?



Is the accident fatal?



How visualizations look with SHAP



1. The scale below the images shows color map for SHAP values.
2. Red pixels means positive contribution to a prediction (i.e removing the pixel lowers accuracy of the model to predict that class)
3. Blue pixels mean negative contribution

Occlusion Sensitivity Mapping

Key characteristics

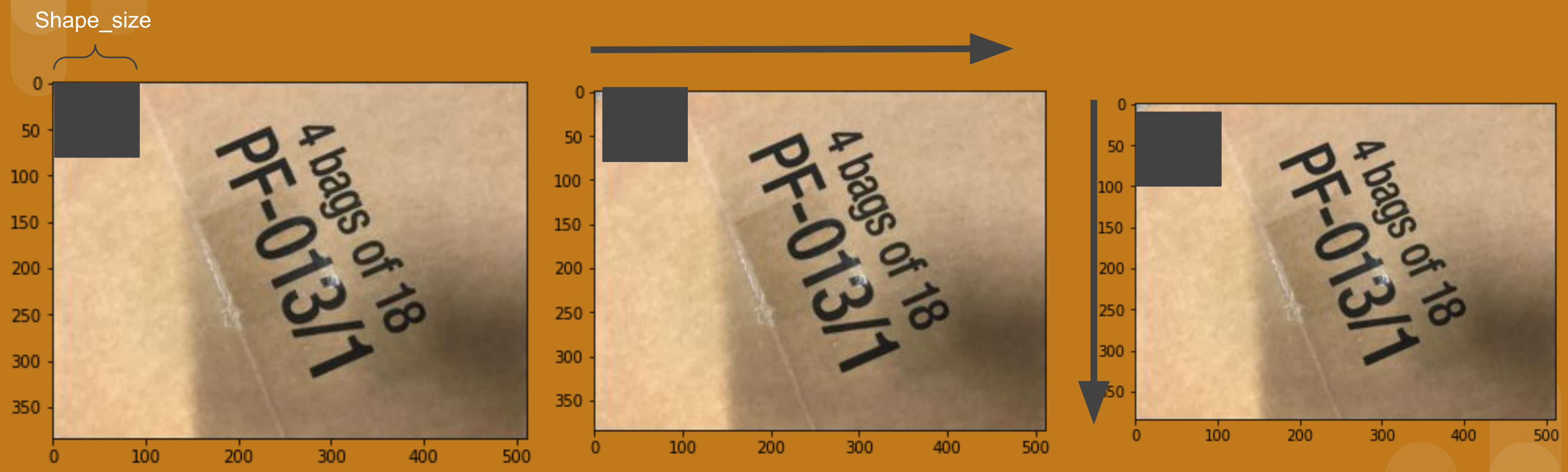
- Model Agnostic
- Local Explanations
- Occlude input data and observe differences in prediction proba for each class.
- Very memory consuming.

Occlusion Sensitivity How it works

Inputs:

- Model
- Data(Image)
- Target class -> True class
- Shape size -> Size of the block to be used for occlusion sensitivity.

Create grey patches with shape_size



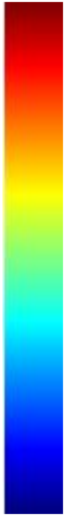
2- Create Sensitivity matrix

- Initialize sensitivity matrix
- Predict all new training datasets created using the input model
- Retrieve probability of target class for each prediction
- Use 1-proba to fill the matrix.(The lower the confidence, the higher the importance of the shaded region).

$$
\begin{array}{c c}
 & \begin{array}{cccc} 1 & 2 & \cdots & n \end{array} \\
\begin{array}{c} 1 \\ 2 \\ 3 \\ \vdots \\ m \end{array} & \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ a_{31} & a_{32} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}
\end{array}
$$

3 - Generate HeatMap

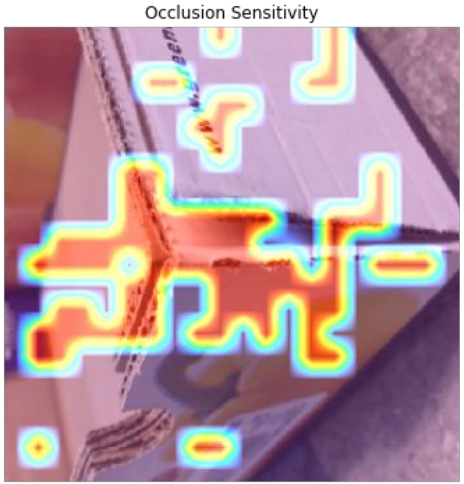
Color Scale



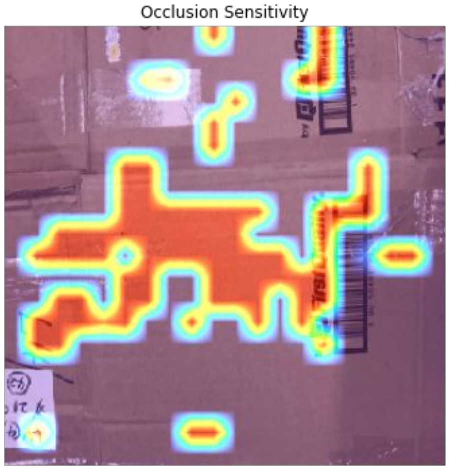
- Resize the sensitivity matrix to original image size.
- Map Sensitivity Map to HeatMap

Examples (tf-explain)

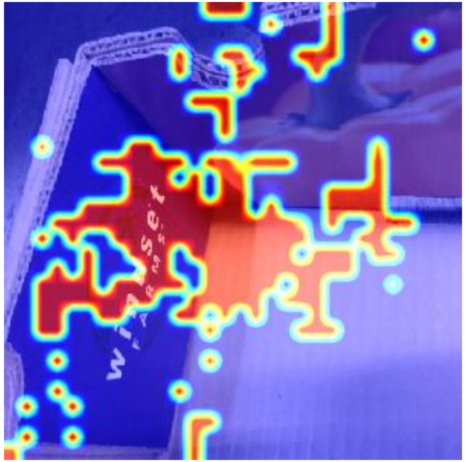
True class: Cardboard
Probability: 78.5%
Classified: Cardboard



True class: Cardboard
Probability: 99%
Classified: Cardboard



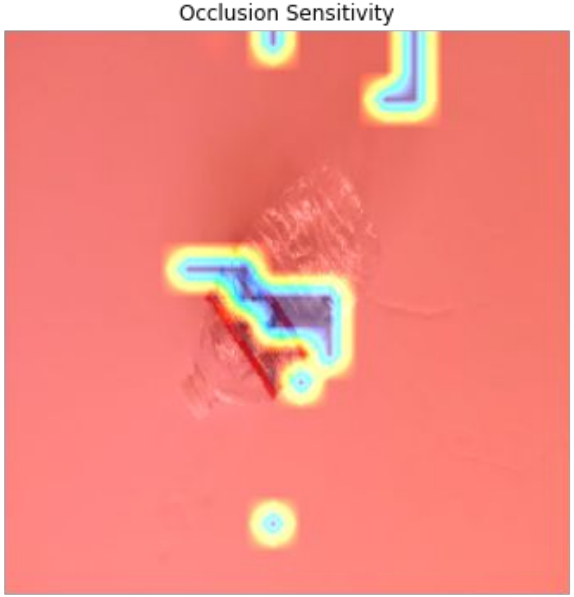
True class: Cardboard
Probability: 87.8%
Classified: Cardboard



Patch_size=10

Examples (tf-explain)

True class: Plastic
Probability: 92%
Classified: Plastic



True class: Plastic
Probability: 99%
Classified: Plastic

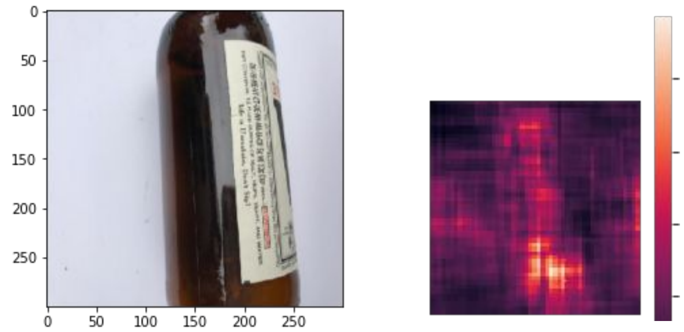
Conclusion tf-explain

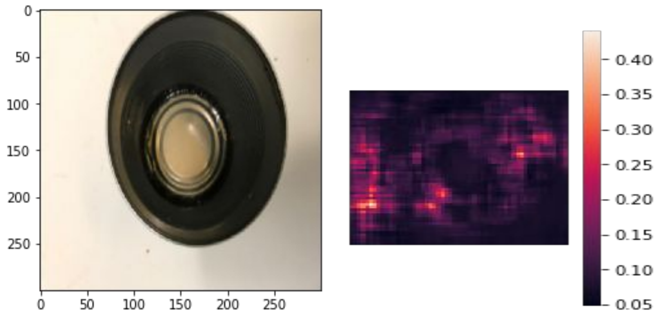
- Tf-explain is not producing results we expected.
- Could be bug in library or bug in our code?
- Let's try to implement a raw, simplified version of occlusion sensitivity

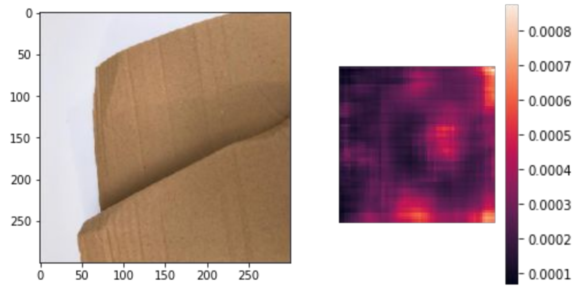
Occlusion sensitivity v2

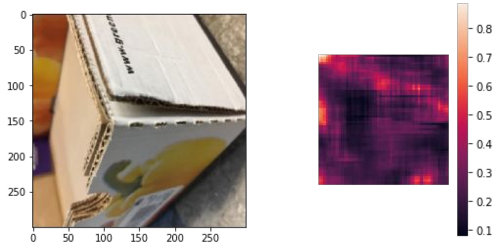
1. Based on https://github.com/oswaldoludwig/Sensitivity-to-occlusion-Keras- (with small changes)
2. Basic idea is very similar to tf-explain
3. Changes in heatmap generation, color scheme.
4. Raw, basic implementation

Examples (raw implementation*)





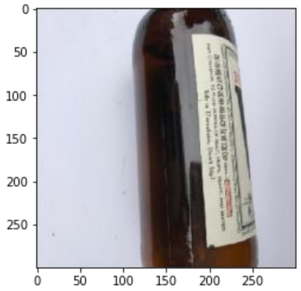




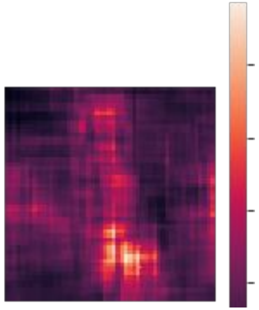
*Results retrieved using 50 pixel grey mask

Mask color problem

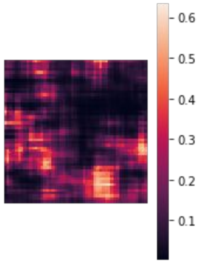
1. The color of the mask can greatly change the generated map depending on colors of the image.
2. Usually grey color is used. What happens if there is grey background, grey metal or cardboard images?



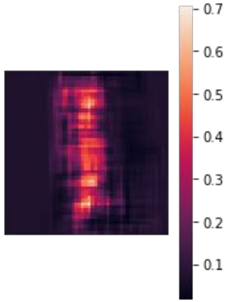
Plain glass wine bottle



Heatmap , grey mask



Heatmap , black mask



Heatmap , white mask

Occlusion sensitivity conclusion

The mask that we use for occlusion can really impact the heatmap generated for our model. There could be many reasons for this:

1. Model could be overfitted due to not enough data.
2. Model could be overfitted due to noise in data, many pictures have just a very big single color background and the trash placed in the middle.
3. Model could be relying to much in the colors?
4. Or maybe this method doesn't make sense to use for our dataset.
5. Best chance we have is to use grey mask which is more neutral color.

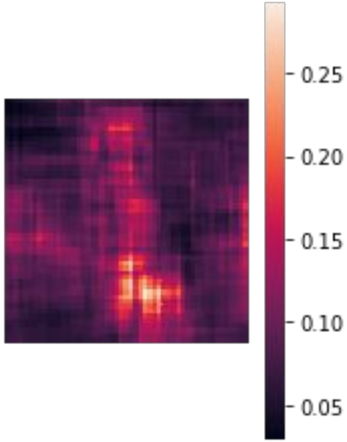
Side by side comparison for same images

Original image

LIME

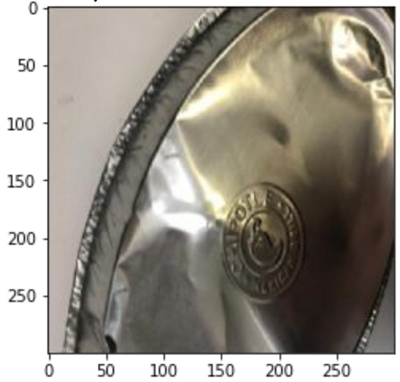
SHAP

Occlusion
Sensitivity v2



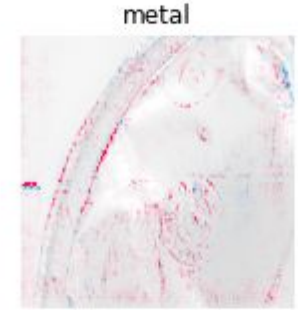
Side by side comparison for same images

Original image (metal, proba = 85%)

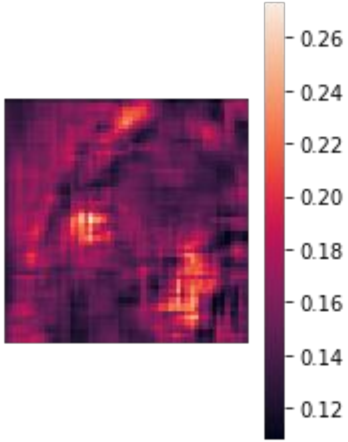


LIME

SHAP

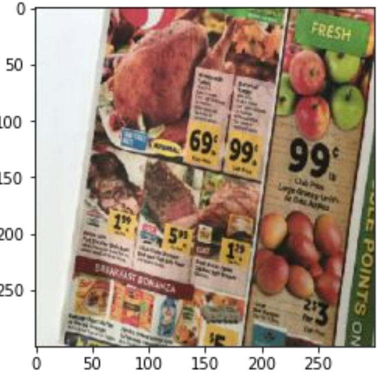


Occlusion Sensitivity v2



Side by side comparison for same images

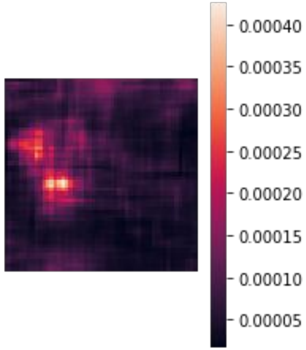
Original image (Paper, proba = 99.99%)



LIME

SHAP

Occlusion Sensitivity v2



Thank you!

Github repository: https://github.com/ekeshi1/explainability_methods_trashnet