

Locally Interpretable Model-Agnostic Explanations

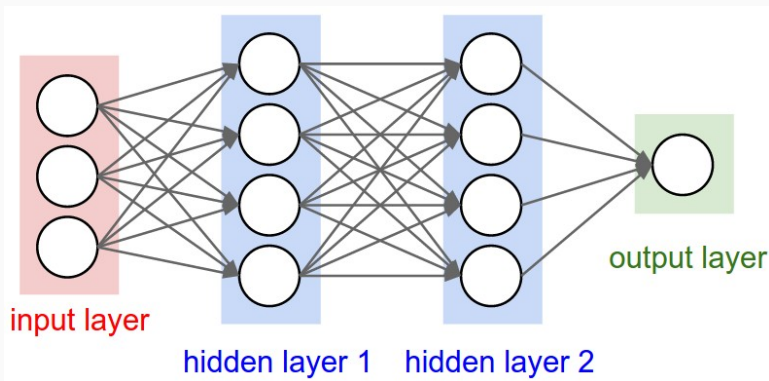
a.k.a LIME a.k.a WTF is my model doing

Alex Hayes

2018-03-29

Motivation

- some models combine data in non-linear ways
- still want to know what's going on



How to explain a model where f is complex?

$X \in \mathbb{R}^{n,p}$ is a data matrix of n points in p space

$f : \mathbb{R}^p \rightarrow \{1, 2, \dots, K\}$ is a classification model

1. Create a binarized and interpretable version of data (called X')
2. Approximate f locally with an interpretable model fit to X'

Multiclass classification treated as K binary classification problems, K the number of classes

Creating interpretable data X'

Original:

```
## # A tibble: 6 x 5
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##         <dbl>         <dbl>         <dbl>         <dbl> <fct>
## 1         5.10         3.50         1.40         0.200 setosa
## 2         4.90         3.00         1.40         0.200 setosa
## 3         4.70         3.20         1.30         0.200 setosa
## 4         4.60         3.10         1.50         0.200 setosa
## 5         5.00         3.60         1.40         0.200 setosa
## 6         5.40         3.90         1.70         0.400 setosa
```

Interpretable:

- Categorical features: one-hot/dummy coding
- Continuous features: effect of value being in a certain range
- Other binary features also good! Presence/absence of shapes in image, sequences in text, etc. . .

Fitting a local approximation: general setting

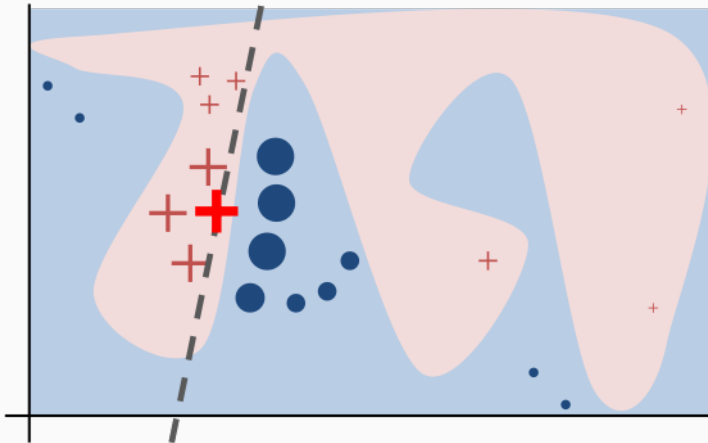
$$\text{explainer} = \operatorname{argmin}_{g \in G} \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

- $f(x)$ is the probability that observation x belongs to a particular class given model f
- G is some class of easily understandable models
- $\mathcal{L}(f, g, \pi_x)$ is a loss function
- π_x is a weighting function that gives higher weights to data close to x
- $\Omega(g)$ is some measure of the complexity of g

The explainer is an easily understood model where all inputs are binary!
It is only locally valid!

The actual estimation process

1. Sample N (typically ~ 5000) data points from X weighted by π_x



2. Turn these into interpretable data vectors X'
3. Fit an understandable model (i.e. LASSO) on X' with response $f(X)$

Get to the darn code already

```
library(lime)
library(caret)

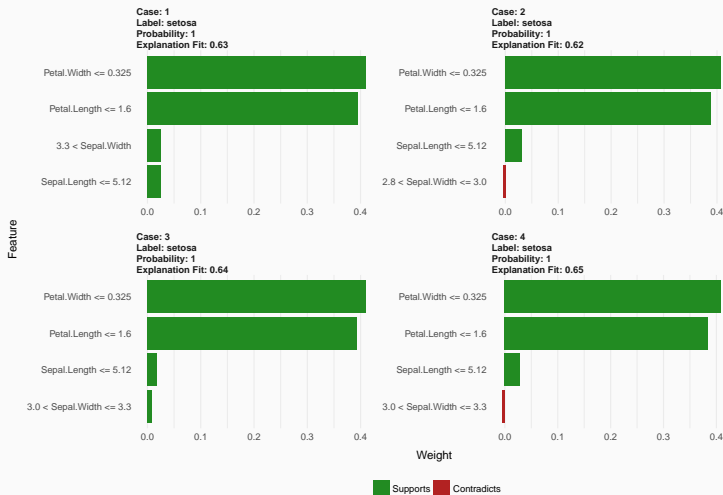
# create train/test sets
iris_train <- slice(iris, 5:n())
iris_test  <- slice(iris, 1:4)

# fit random forest
model <- train(Species ~ ., iris_train,
               method = 'rf')

# fit explainer object
explainer <- lime(iris_train, model)
```

```
# explain observations!  
explanation <- explain(iris_test,  
                      explainer,  
                      n_labels = 1,  
                      n_features = 4)  
  
# plot_features(explanation) -- results on next slide
```


Visual explanations



What is this good for?

- you now have an approximation g of f
- you can understand g

However:

- you have no idea how good the approximation is
- *key*: g **does not** explain what data is causing what response in f

My take: use for diagnostics and intuition only.

- is my model doing something really stupid?
- is there an obvious bias in my model?

lime R just works with models from:

- caret
- mlr
- xgboost
- h2o
- keras

Easy to extend to other packages. Can also explain image and text classifications!

Anything you can send into a neural net and get class probs out of you can explain!

Questions?

The original paper: “Why Should I Trust You?”: Explaining the Predictions of Any Classifier

Generalization of LIME: A Unified Approach to Interpreting Model Predictions

@alexpghayes on Twitter
alexpghayes@gmail.com

Sparse Linear Explanations

Algorithm 1 Sparse Linear Explanations using LIME

Require: Classifier f , Number of samples N

Require: Instance x , and its interpretable version x'

Require: Similarity kernel π_x , Length of explanation K

$\mathcal{Z} \leftarrow \{\}$

for $i \in \{1, 2, 3, \dots, N\}$ **do**

$z'_i \leftarrow \text{sample_around}(x')$

$\mathcal{Z} \leftarrow \mathcal{Z} \cup \langle z'_i, f(z_i), \pi_x(z_i) \rangle$

end for

$w \leftarrow \text{K-Lasso}(\mathcal{Z}, K) \triangleright$ with z'_i as features, $f(z)$ as target

return w

K-Lasso: Use RMSE weighted by π_x as loss. Select top K predictors with LASSO. Put these into OLS for final explainer.