Interpreting machine learning models
or how to turn a random forest into a white box

Ando Saabas
About me

• Senior applied scientist at Microsoft,
  • Using ML and statistics to improve call quality in Skype
  • Various projects on user engagement modelling, Skype user graph analysis, call reliability modelling, traffic shaping detection

• Previously, working on programming logics with Tarmo Uustalu
Machine learning and model interpretation

• Machine learning studies algorithms that learn from data and make predictions
• Learning algorithms are about *correlations* in the data

• In contrast, in data science and data mining, understanding *causality* is essential
• Applying domain knowledge requires understanding and interpreting models
Usefulness of model interpretation

• Often, we need to understand individual predictions a model is making.

For example a model may

• Recommend a treatment for a patient or estimate a disease to be likely. The doctor needs to understand the reasoning.
• Classify a user as a scammer, but the user disputes it. The fraud analyst needs to understand why the model made the classification.
• Predict that a video call will be graded poorly by the user. The engineer needs to understand why this type of call was considered problematic.
Usefulness of model interpretation cont.

• Understanding differences on a dataset level.
  • Why is a new software release receiving poorer feedback from customers when compared to the previous one?
  • Why are grain yields in one region higher than the other?

• Debugging models. A model that worked earlier is giving unexpected results on newer data.
Algorithmic transparency

• Algorithmic transparency becoming a requirement in many fields

• French Conseil d’État (State Council’s) recommendation in „Digital technology and fundamental rights“(2014): *Impose to algorithm-based decisions a transparency requirement, on personal data used by the algorithm, and the general reasoning it followed.*

• Federal Trade Commission (FTC) Chair Edith Ramirez: *The agency is concerned about ‘algorithmic transparency’../”* (Oct 2015). FTC Office of Technology Research and Investigation started in March 2015 to tackle algorithmic transparency among other issues
Interpretable models

- Traditionally, two types of (mainstream) models considered when interpretability is required
- Linear models (linear and logistic regression)
  - \( Y = a + b_1 x_1 + \cdots + b_n x_n \)
  - heart_disease = 0.08*tobacco + 0.043*age + 0.939*famhist + ...
    (from Elements of Statistical Learning)
- Decision trees
**Example: heart disease risk prediction**

### CORONARY DISEASE RISK PREDICTION SCORE SHEET FOR MEN BASED ON TOTAL CHOLESTEROL LEVEL

<table>
<thead>
<tr>
<th>Step 1</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Years</td>
</tr>
<tr>
<td></td>
<td>30-34</td>
</tr>
<tr>
<td></td>
<td>35-39</td>
</tr>
<tr>
<td></td>
<td>40-44</td>
</tr>
<tr>
<td></td>
<td>45-49</td>
</tr>
<tr>
<td></td>
<td>50-54</td>
</tr>
<tr>
<td></td>
<td>55-59</td>
</tr>
<tr>
<td></td>
<td>60-64</td>
</tr>
<tr>
<td></td>
<td>65-69</td>
</tr>
<tr>
<td></td>
<td>70-74</td>
</tr>
</tbody>
</table>

### Step 2: Total Cholesterol

<table>
<thead>
<tr>
<th>Total Cholesterol (mg/dL)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;160</td>
<td>-3</td>
</tr>
<tr>
<td>160-199</td>
<td>0</td>
</tr>
<tr>
<td>200-239</td>
<td>1</td>
</tr>
<tr>
<td>240-279</td>
<td>2</td>
</tr>
<tr>
<td>≥280</td>
<td>3</td>
</tr>
</tbody>
</table>

### Step 3: HDL - Cholesterol

<table>
<thead>
<tr>
<th>HDL - Cholesterol (mg/dL)</th>
<th>Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;35</td>
<td>2</td>
</tr>
<tr>
<td>35-44</td>
<td>1</td>
</tr>
<tr>
<td>45-49</td>
<td>0</td>
</tr>
<tr>
<td>50-59</td>
<td>0</td>
</tr>
<tr>
<td>≥60</td>
<td>-2</td>
</tr>
</tbody>
</table>

### Step 4: Blood Pressure

<table>
<thead>
<tr>
<th>Blood Pressure</th>
<th>Systolic</th>
<th>Diastolic</th>
</tr>
</thead>
</table>

### Step 7: Adding up the points

<table>
<thead>
<tr>
<th>Adding up the points</th>
<th>Age</th>
<th>Total Cholesterol</th>
<th>HDL Cholesterol</th>
<th>Blood Pressure</th>
<th>Diabetes</th>
<th>Smoker</th>
<th>Point Total</th>
</tr>
</thead>
</table>

### Step 8: Determine CHD risk from point total

<table>
<thead>
<tr>
<th>CHD Risk</th>
<th>Point Total</th>
<th>10 Yr CHD Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;1</td>
<td>2%</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3%</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4%</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>5%</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>7%</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>8%</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>10%</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>13%</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>16%</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>20%</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>25%</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>31%</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>37%</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>45%</td>
<td></td>
</tr>
<tr>
<td>≥14</td>
<td>≥53%</td>
<td></td>
</tr>
</tbody>
</table>

*From National Heart, Lung and Blood Institute.*
Linear models have drawbacks

- Underlying data is often non-linear
Tackling non-linearity

• Feature binning: create new variables for various intervals of input features
  • For example, instead of feature $x$, you might have
    • $x_{\text{between}_0\_\text{and}_1}$
    • $x_{\text{between}_1\_\text{and}_2}$
    • $x_{\text{between}_2\_\text{and}_4}$ etc
  • Potentially massive increase in number of features

• Basis expansion (non-linear transformations of underlying features)
  $Y = 2x_1 + x_2 - 3x_3$
  vs
  $Y = 2x_1^2 - 3x_1 - \log(x_2) + \sqrt{x_2}x_3 + ...$

In both cases performance is traded for interpretability
Decision trees

- Decision trees can fit to non-linear data
- They work well with both categorical and continuous data, classification and regression
- Easy to understand
Or are they?

(Small part of) default decision tree in scikit-learn. Boston housing data
500 data points
14 features
Decision trees

- Decision trees are understandable only when they are (very) small
  - Tree of depth $n$ has up to $2^n$ leaves and $2^n - 1$ internal nodes. With depth 20, a tree can have up to 1048576 leaves
  - Previous slide had <200 nodes

- Additionally, decision trees are high variance method – low generalization, tend to overfit
Random forests

• Can learn non-linear relationships in the data well
• Robust to outliers
• Can deal with both continuous and categorical data
• Require very little input preparation (see previous three points)
• Fast to train and test, trivially parallelizable
• High accuracy even with minimal meta-optimization

• Considered to be a **black box** that is difficult or impossible to interpret
Random forests as a black box

- Consist of a large number of decision trees (often 100s to 1000s)
- Trained on bootstrapped data (sampling with replacement)
- Using random feature selection
Random forests as a black box

• "Black box models such as random forests can't quantify the impact of each predictor to the predictions of the complex model”, in PRICAI 2014: Trends in Artificial Intelligence

• "Unfortunately, the random forest algorithm is a black box when it comes to evaluating the impact of a single feature on the overall performance". In Advances in Natural Language Processing 2014

• “(Random forest model) is close to a black box in the sense that it uses 810 features ./../ reduction in the number of features would allow an operator to study individual decisions to have a rough idea how the global decision could have been made”. In Advances in Data Mining: Applications and Theoretical Aspects: 2014
Understanding the model vs the predictions

• Keep in mind, we want to understand why a particular decision was made. Not necessarily every detail of the full model

• As an analogy, we don’t need to understand how a brain works to understand why a person made a particular a decision: simple explanation can be sufficient

• Ultimately, as ML models get more complex and powerful, hoping to understand the models themselves is doomed to failure

• We should strive to make models explain their decisions
Turning the black box into a white box

• In fact, random forest predictions can be explained and interpreted, by decomposing predictions into mathematically exact feature contributions

• Independently of the
  • number of features
  • number of trees
  • depth of the trees
Revisiting decision trees

• Classical definition (from *Elements of statistical learning*)

\[ dt(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \]

• Tree divides the feature space into M regions \( R_m \) (one for each leaf)

• Prediction for feature vector \( x \) is the constant \( c_m \) associated with region \( R_m \), the vector \( x \) belongs to
Example decision tree – predicting apartment prices

- **Rooms < 3**
  - **Yes**
    - **Crime rate < 3**
      - **Built_year > 2008**
        - 45,000
      - **30,000**
    - **55,000**
  - **No**
    - **Floor < 2**
      - **Crime_rate < 5**
        - 52,000
      - 70,000
Estimating apartment prices

Assume an apartment [2 rooms; Built in 2010; Neighborhood crime rate: 5]
We walk the tree to obtain the price
Estimating apartment prices

[2 rooms; Built in 2010; Neighborhood crime rate: 5]
Estimating apartment prices

[2 rooms; **Built in 2010**; Neighborhood crime rate: 5]
Estimating apartment prices

Prediction: 35,000

Path taken: Rooms < 3, Built_year > 2008, Crime_rate < 3

[2 rooms; Built in 2010; Neighborhood crime rate: 5]
Operational view

• Classical definition ignores the operational aspect of the tree.

• There is a decision path through the tree

• All nodes (not just the leaves) have a value associated with them
Internal values

- All internal nodes have a value associated with them
- At depth 0, prediction would simply be the dataset mean (assuming we want to minimize squared loss)
- When training the tree, we keep expanding it, obtaining new values
Internal values

Rooms < 3

Yes: 48,000

No: 57,000

Yes: 33,000
Internal values
Internal values

Rooms < 3
  Yes
  Built_year > 2008
    Crime_rate < 3
      40,000
      45,000
    33,000
  48,000
  No
    Floor < 2
      55,000
      57,000
    Crime_rate < 5
      60,000
      52,000
Operational view

• All nodes (not just the leaves) have a value associated with them
• Each decision along the path contributes something to the final outcome
• A feature is associated with every decision
• We can compute the final outcome in terms of feature contributions
Estimating apartment prices revisited
Estimating apartment prices

\[ E(price) = 48,000 \]
Estimating apartment prices

\[ E(price|\text{rooms} < 3) = 33,000 \]

Price = 48,000 − 15,000(\text{Rooms})

[2 rooms]
Estimating apartment prices

Price = 48,000 – 15,000(Rooms) + 7,000(Built_year)  
[Built 2010]
Estimating apartment prices

Price = 48,000 – 15,000(Rooms) + 7,000(Built_year) – 5,000(Crime_rate)

[Crime rate 5]
Estimating apartment prices

Price = 48,000 – 15,000(Rooms) + 7,000(Built_year) – 5,000(Crime_rate) = 35,000
We can define the decision tree in terms of the bias and contribution from each feature.

\[
dt(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \quad \Rightarrow \quad dt(x) = \text{bias} + \sum_{i=1}^{N} \text{contr}(i, x)
\]

Similar to linear regression (prediction = bias + feature_1 contribution + ... + feature_n contribution), but on a prediction level, not model level.

Does not depend on the size of the tree or number of features.

Works equally well for
  - Regression and classification trees
  - Multivalued and multilabel data
Deeper inspections

• We can have more fine grained definition in addition to pure feature contributions
  • Separate negative and positive contributions
  • Contribution from decisions (floor = 1 \(\rightarrow\) -15000)
  • Contribution from interactions (floor == 1 & has_terrace \(\rightarrow\) 3000)
  • etc

• Number of features typically not a concern because of the long tail. In practice, top 10 features contribute the vast majority of the overall deviation from the mean
From decision trees to random forests

- Prediction of a random forest is the average of the predictions of individual trees

- From distributivity of multiplication and associativity of addition, random forest prediction can be defined as the average of bias term of individual trees + sum of averages of each feature contribution from individual trees

\[
RF(x) = \frac{1}{J} \sum_{j=1}^{J} dt_j(x)
\]

\[
RF(x) = \frac{1}{J} \sum_{j=1}^{J} bias_j(x) + \left( \frac{1}{J} \sum_{j=1}^{J} contr_j(1, x) \right) + \cdots + \left( \frac{1}{J} \sum_{j=1}^{J} contr_j(n, x) \right)
\]

- prediction = bias + feature_1 contribution + ... + feature_n contribution
Random forest interpretation in Scikit-Learn

• Path walking in general no supported by ML libraries
• Scikit-Learn: one of the most popular Python (and overall) ML libraries
• Patch since 0.17 (released Nov 8 2015) to include values/predictions at each node: allows walking the tree paths and collecting values along the way
• Treeinterpreter library for decomposing the predictions
  • https://github.com/andosa/treeinterpreter
  • pip install treeinterpreter
• Supports both decision tree and random forest classes, regression and classification
Using treeinterpreter

• Decomposing predictions is a one-liner

from treeinterpreter import treeinterpreter as ti
rf = RandomForestRegressor()
rf.fit(trainX, trainY)

prediction, bias, contributions = ti.predict(rf, testX)
# instead of prediction = rf.predict(testX)

#prediction == bias + contributions
assert(np.allclose(prediction, bias + np.sum(contributions, axis=1)))
Decomposing a prediction – boston housing data

```python
prediction, bias, contributions = ti.predict(rf, boston.data)

>> prediction[0]
30.69
>> bias[0]
25.588

>> sorted(zip(contributions[0], boston.feature_names),
   key=lambda x: -abs(x[0]))

[(4.3387165697195558, 'RM'),
(-1.0771391053864874, 'TAX'),
(1.0207116129073213, 'LSTAT'),
(0.38890774812797702, 'AGE'),
(0.38381481481481539, 'ZN'),
(-0.10397222222222205, 'CRIM'),
(-0.091520697167756987, 'NOX')
...]
```
Caveats

• The method assumes that features are actually interpretable in the first place.
  • This does not always hold: e.g. pixels in image classification
  • Can be overcome via postprocessing

• In presence of strong correlations in the input data, true causal features can be buried under non-causal but correlated features. Domain knowledge and feature tuning required in this case
Conclusions

• Model interpretation can be very beneficial for ML and data science practitioners in many tasks

• No need to understand the full model in many/most cases: explanation of decisions sufficient

• Random forests can be turned from black box into a white box
  • Each prediction can decomposed into bias and feature contribution terms

• Python library available for scikit-learn at https://github.com/andosa/treeinterpreter or pip install treeinterpreter