COMPSCI 389

Introduction to Machine Learning

Days: Tu/Th.  Time: 2:30 – 3:45  Building: Morrill 2  Room: 222

Topic 3.0: Models, Algorithm Template, Nearest Neighbor

Prof. Philip S. Thomas (pthomas@cs.umass.edu)
Review

• Input output pair \((X, Y)\)
  • \(X\): Input, features, attributes, covariates, or predictors
    • Numerical (discrete/continuous), categorical (nominal/ordinal), etc.
  • \(Y\): Output, label, or target
    • Regression: \(Y\) is continuous.
    • Classification: \(Y\) is discrete

• Data set: \((X_i, Y_i)_{i=1}^{n}\)

• Query: An additional input \(X\)

• Goal: Predict the label \(Y\) associated with \(X\).
Models (Supervised Learning)

- A **model** is a mechanism that maps input data to predictions.
- **(Offline) ML algorithms** take data sets as input and produce models as output.

**Online** ML algorithms can receive data over time, improving their models as more data becomes available.

### ML Algorithm

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Model</th>
<th>Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>physics</td>
<td>biolgy</td>
<td>history</td>
</tr>
<tr>
<td>527.66</td>
<td>443.82</td>
<td>545.88</td>
</tr>
<tr>
<td>527.66</td>
<td>443.82</td>
<td>545.88</td>
</tr>
</tbody>
</table>

A query can be one or more feature vectors.

Predictions are given for each feature vector in the query.
Training/Fitting

• ML algorithms could take a data set and query at the same time and output a prediction for the query.
  • Each time a new query is given, the algorithm re-processes the entire data set.
• Idea: More efficient to preprocess the data set, computing relevant statistics and quantities.
  • Given a query, the algorithm might reference the statistics and quantities it computed without re-referencing the data set at all!
• This pre-processing of the data set is called **training**.
  • Sometimes: “Training the model”
  • Sometimes: “Fitting the model to data”
  • Sometimes: “Pre-processing data”
Scikit-Learn Models

• Scikit-Learn is a popular ML library in python.
• It has objects called “models”.
  • These “models” are more than just models – they are complete ML algorithms.
Scikit-Learn Models

• Scikit-Learn models implement the functions:
  • `fit(self, X, y)`: The function for fitting the model to the data (training the model given the data / preprocessing the data).
    • `X`: A 2D array-like structure (e.g., DataFrame) representing the features. Each row is a point and each column is a feature.
    • `y`: A 1D array-like structure (e.g., Series) representing the target values
    • Returns `self` to simplify chaining together operations.
  • `predict(self, X)`: The function for producing predictions given queries.
    • `X`: A 2D array-like structure representing the data for which predictions are to be made. Each row is a sample and each column is a feature.
    • Returns a numpy array of predicted labels/values.

• **Note**: Ideally fit and predict are compatible with `X` and `y` being DataFrames or numpy arrays.
from sklearn.base import BaseEstimator
import numpy as np

class CustomMLAlgorithm(BaseEstimator):
    def __init__(self, param1=1, param2=2):
        # Initialization code
        self.param1 = param1
        self.param2 = param2

    def fit(self, X, y):
        # Training code
        # Implement your training algorithm here
        return self

    def predict(self, X):
        # Prediction code
        # Implement your prediction algorithm here
        return np.zeros(len(X))

• Given data set (X,y) and query:
  model = CustomMLAlgorithm()
  model.fit(X,y)
  predictions = model.predict(query)
Nearest Neighbor

- A particularly simple yet effective ML algorithm based on the core idea: 
  \textit{When presented with a query, find the data point (row) that is most similar to the query and give the label associated with this most-similar point as the prediction.}

- We can map this to fit/predict functions:
  - \texttt{fit}: Store the data
  - \texttt{predict}: For each query row do the following
    - Loop over each row in the training data, computing the Euclidean distance between the query and the row.
    - Create an array holding the labels from the rows with the smallest distance to the query feature vector (often just one element).
    - Return an arbitrary (e.g., random) element of the array.
### Query:

<table>
<thead>
<tr>
<th>physics</th>
<th>biology</th>
<th>history</th>
<th>English</th>
<th>geography</th>
<th>literature</th>
<th>Portuguese</th>
<th>math</th>
<th>chemistry</th>
<th>distance</th>
<th>gpa</th>
</tr>
</thead>
<tbody>
<tr>
<td>622.6</td>
<td>491.56</td>
<td>439.93</td>
<td>707.64</td>
<td>663.65</td>
<td>557.09</td>
<td>711.37</td>
<td>731.31</td>
<td>509.8</td>
<td>1.33333</td>
<td></td>
</tr>
<tr>
<td>538</td>
<td>490.58</td>
<td>406.59</td>
<td>529.05</td>
<td>532.28</td>
<td>447.23</td>
<td>527.58</td>
<td>379.14</td>
<td>488.64</td>
<td>2.98333</td>
<td></td>
</tr>
<tr>
<td>455.18</td>
<td>440</td>
<td>570.86</td>
<td>417.54</td>
<td>453.53</td>
<td>425.87</td>
<td>475.63</td>
<td>476.11</td>
<td>407.15</td>
<td>1.97333</td>
<td></td>
</tr>
<tr>
<td>756.91</td>
<td>679.62</td>
<td>531.28</td>
<td>583.63</td>
<td>534.42</td>
<td>521.4</td>
<td>783.76</td>
<td>588.26</td>
<td>529.04</td>
<td>2.53333</td>
<td></td>
</tr>
<tr>
<td>584.54</td>
<td>649.84</td>
<td>637.43</td>
<td>609.06</td>
<td>670.46</td>
<td>515.38</td>
<td>572.52</td>
<td>581.25</td>
<td>529.04</td>
<td>1.58667</td>
<td></td>
</tr>
<tr>
<td>325.99</td>
<td>466.74</td>
<td>597.06</td>
<td>554.43</td>
<td>535.77</td>
<td>717.03</td>
<td>477.6</td>
<td>503.82</td>
<td>422.92</td>
<td>1.66667</td>
<td></td>
</tr>
<tr>
<td>622.6</td>
<td>587.04</td>
<td>598.85</td>
<td>603.32</td>
<td>690.7</td>
<td>652.86</td>
<td>533.05</td>
<td>755.3</td>
<td>628.73</td>
<td>3.72333</td>
<td></td>
</tr>
<tr>
<td>527.65</td>
<td>559.99</td>
<td>758.37</td>
<td>669.71</td>
<td>645.62</td>
<td>648.67</td>
<td>539.23</td>
<td>470.78</td>
<td>486.13</td>
<td>3.08333</td>
<td></td>
</tr>
<tr>
<td>647.64</td>
<td>687.83</td>
<td>630.61</td>
<td>613.95</td>
<td>557.43</td>
<td>739.94</td>
<td>557.27</td>
<td>557.14</td>
<td>632.54</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
See IPython Notebook (3.1)
Intermission

• Class will resume in 5 minutes.
• Feel free to:
  • Stand up and stretch.
  • Leave the room.
  • Talk to those around you.
• Write a question on a notecard and add it to the stack at the front of the room.