The Machine Learning Audit

Andrew Clark, Principal Machine Learning Auditor
Capital One
Overview

❖ What is a Machine Learning?
❖ Why is it important?
❖ Why do we need machine learning audits?
❖ What exactly is a machine learning audit?
❖ What would a machine learning audit entail?
❖ Full-length example using the CRISP-DMA framework
What is Machine Learning?

- A computer recognizing patterns without having to be explicitly programmed.
AI

Machine Learning

Deep Learning
THE DATA SCIENCE HIERARCHY OF NEEDS

1. COLLECT
   - Instrumentation, logging, sensors, external data, user generated content

2. MOVE/STORE
   - Reliable data flow, infrastructure, pipelines, ETL, structured and unstructured data storage

3. EXPLORE/TRANSFORM
   - Cleaning, anomaly detection, prep

4. AGGREGATE/LABEL
   - Analytics, metrics, segments, aggregates, features, training data

5. LEARN/OPTIMIZE
   - A/B testing, experimentation, simple ML algorithms

6. AI, DEEP LEARNING

@mrogati
Why is Machine Learning Important?

- Disrupting business. Example ML powered businesses disrupted Blockbuster, Taxis, etc.
- Revolutionizing existing business models. Predictive maintenance in manufacturing, retailing, credit card fraud detection, loan underwriting.
- One of the key technologies in driving economic growth.
- One of the most talked about but least understood topics in modern discourse. e.x. “Facebook shuts down robots after they invent their own language” (The Telegraph August 1, 2017) and “Elon Musk: regulate AI to combat 'existential threat' before it's too late” (The Guardian July 17, 2017).
- Sensational stories are clickbait.
What Machine Learning is not:

- Magic
- Going to take your job (for the majority of professionals)
- Always the best tool for the job
Why do we need machine learning audits?

- With algorithms increasingly dictating our lives, how do we know that they are operating as intended?
- e.x. Weapons of Math Destruction by Cathy O'Neil
- Some believe the EU General Data Protection Regulation act provides a “Right to Explanation”, although this is not explicitly stated and is untested in the courts.
What exactly is a machine learning audit?

- Examination of the purpose, process, execution, and monitoring of a machine learning model ‘in the wild’.

- As assurance professionals, how do we know that the model is doing what it should be doing? What is the risk to the business?

- Data Science is a new discipline, without the formal rigor and mature of processes that exist in other disciplines. Statistics is a profession that has been around for years, yet there are so many issues with the peer review process of statistics, and their models aren’t as complicated!
What would a machine learning audit entail?

- Understand the business use case.
- Model integration into existing architecture.
- Potential regulatory or risk constraints
- “Data Sciencey stuff” – i.e.
  - How was the test data obtained?
  - How was it cleaned?
  - How was the feature engineering conducted?
  - How was the specific algorithm decided upon?
  - Are there correction cascades?
  - How was the model evaluated?
  - What was the process to prevent overfitting, etc.
- Is the model accomplishing what the business wanted it to accomplish?
Introducing the CRISP-DMA framework

- Framework written by yours truly that extends the industry standard data mining framework, CRISP-DM to auditing machine learning implementations.
- Leverages that existing, eight, iterative steps of the CRISP-DM model:
  - Business Understanding
  - Data Understanding
  - Data Preparation
  - Modeling
  - Evaluation
  - Deployment
Business Understanding

- What is the goal of the algorithm?
- Have models been used in this use case before?
- What attributes, i.e. temperature, humidity, etc., have been identified by the business as key factors for deriving the desired decision in the given use case?
- Are there any regulatory constraints or considerations of which to be aware?
Data Understanding

❖ What dataset[s] was utilized to train the model?
❖ What dataset[s] is utilized for production prediction?
❖ Where did the data set[s] identified in 1,2 originate? I.e. web scrapped data, log files, relational databases.
❖ Are all of the input variables in the same format? I.e. miles or kilometers.
❖ Have the correlations and covariances been examined?
Data Preparation

❖ How was the data cleaned?
❖ If supervised learning was used, how was the training dataset created?
❖ Were standard software development techniques used for the ETL process for production models?
❖ How was the data scaled?
❖ How were the variables selected? Was an automated variable selection technique utilized?
❖ What process was used to separate the data into train and test sets? Was care taken to avoid peaking at the test set?
Modeling

- What was the thought process behind choosing algorithm[s] for the model?
- What steps were used to guard against overfitting?
- What process was used to optimize the chosen algorithm?
- Was the algorithm coded from scratch or was a standard library used? If so, what are the license terms of the library?
- What type of version control was utilized?
What metrics were used to evaluate the model?

What process and metrics are in place to monitor the continued accuracy and stability of the model?

Create a mock dataset that covers all of the relevant assumptions and run the results through the algorithm to test that it is operating as intended.
Deployment

- How was the model moved to production? Was it rewritten by the engineering team, or does it rely on an API, etc., (if it was rewritten, a code review for accuracy should be performed).

- Is the model accomplishing what the business wanted it to accomplish?
Raspberry Pi Machine Learning Weather Prediction - A simple example
Architecture Diagram

AWS EC2 → AWS RDS

Raspberry Pi 3 with DHT22 Humidity/Temperature sensor

Rain in the next 30 mins?

If yes, email alert
Raspberry Pi readings and actual weather

### Weather

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-08-15 20:57:41</td>
<td>78.8</td>
<td>83</td>
<td>Clouds</td>
</tr>
<tr>
<td>2017-08-15 21:03:41</td>
<td>78.8</td>
<td>83</td>
<td>Clouds</td>
</tr>
<tr>
<td>2017-08-15 21:09:41</td>
<td>78.8</td>
<td>83</td>
<td>Clouds</td>
</tr>
<tr>
<td>2017-08-15 21:15:41</td>
<td>78.8</td>
<td>83</td>
<td>Clouds</td>
</tr>
<tr>
<td>2017-08-15 21:21:41</td>
<td>78.8</td>
<td>83</td>
<td>Clouds</td>
</tr>
</tbody>
</table>

### Telemetry

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Temperature</th>
<th>Humidity</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-08-15 20:30:51</td>
<td>76.0</td>
<td>47.0</td>
</tr>
<tr>
<td>2017-08-15 20:36:52</td>
<td>76.0</td>
<td>47.0</td>
</tr>
<tr>
<td>2017-08-15 20:42:53</td>
<td>76.0</td>
<td>46.0</td>
</tr>
<tr>
<td>2017-08-15 20:48:55</td>
<td>76.0</td>
<td>46.0</td>
</tr>
<tr>
<td>2017-08-15 20:54:56</td>
<td>76.0</td>
<td>47.0</td>
</tr>
</tbody>
</table>
Aggregate readings to one average reading every thirty minutes

```python
In [18]:
# Create new dataframe that resamples to every 30 minutes
TelmetryHalfHour = pd.DataFrame()
TelmetryHalfHour['Temperature'] = Telemetry.Temperature.resample('30T').mean()
#TelmetryHalfHour['Pressure'] = Telemetry.Pressure.resample('30T').mean()
TelmetryHalfHour['Humidity'] = Telemetry.Humidity.resample('30T').mean()

In [19]:
TelmetryHalfHour.tail()
```

```
Out[19]:
   Timestamp  Temperature  Humidity
Timestamp
2017-08-15 18:30:00  76.4      48.4
2017-08-15 19:00:00  76.4      47.6
2017-08-15 19:30:00  77.2      47.4
2017-08-15 20:00:00  76.4      47.4
2017-08-15 20:30:00  76.0      46.6
```
Aggregation cont.

```python
# Shift the status back 30 minutes to allow for prediction
CombinedDF['Status_Shift'] = CombinedDF.Status.shift(-1)
```

```
In [34]: CombinedDF.tail()
```

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Pie_Temp</th>
<th>Pie_Hum</th>
<th>Weather_Temp</th>
<th>Weather_Hum</th>
<th>Status</th>
<th>Status_Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>2017-08-15 18:30:00</td>
<td>76.4</td>
<td>48.4</td>
<td>88.16</td>
<td>57.4</td>
<td>Clouds</td>
<td>Rain</td>
</tr>
<tr>
<td>2017-08-15 19:00:00</td>
<td>76.4</td>
<td>47.6</td>
<td>78.80</td>
<td>83.0</td>
<td>Rain</td>
<td>Rain</td>
</tr>
<tr>
<td>2017-08-15 19:30:00</td>
<td>77.2</td>
<td>47.4</td>
<td>78.80</td>
<td>83.0</td>
<td>Rain</td>
<td>Rain</td>
</tr>
<tr>
<td>2017-08-15 20:00:00</td>
<td>76.4</td>
<td>47.4</td>
<td>78.80</td>
<td>83.0</td>
<td>Rain</td>
<td>Rain</td>
</tr>
<tr>
<td>2017-08-15 20:30:00</td>
<td>76.0</td>
<td>46.6</td>
<td>78.80</td>
<td>83.0</td>
<td>Rain</td>
<td>NaN</td>
</tr>
</tbody>
</table>
Convert the status to 1 if the status is rain or thunderstorm, 0 otherwise

```python
In [35]:
# Use list comprehension for labeling the column to predict
CombinedDF['Rain?'] = [1 if x == 'Rain' or x == 'Thunderstorm' else 0 for x in CombinedDF['Status_Shift']]  

In [36]:
# remove the status column
del CombinedDF['Status']
del CombinedDF['Status_Shift']
CombinedDF.tail()

Out[36]:
   Timestamp   Pie_Temp  Pie_Hum  Weather_Temp  Weather_Hum    Rain?
0  2017-08-15 18:30:00    76.4    48.4        88.16       57.4      1
1  2017-08-15 19:00:00    76.4    47.6        78.80       83.0      1
2  2017-08-15 19:30:00    77.2    47.4        78.80       83.0      1
3  2017-08-15 20:00:00    76.4    47.4        78.80       83.0      1
4  2017-08-15 20:30:00    76.0    46.6        78.80       83.0      0
```
Split the data into training and test sets

```python
from sklearn.model_selection import ShuffleSplit

# predict the X and y data:
y = CombinedDF['Rain?'].values # return the labels we want
deleted = CombinedDF['Rain?'] # remove the win label
X = CombinedDF.values # use everything else to predict

# split the data into training and test sets using sklearn's ShuffleSplit
cv_object = ShuffleSplit(n_splits=3, test_size=.20, random_state=10)

for train_index, test_index in cv_object.split(X):
    # New variables
    X_train = X[train_index]
    y_train = y[train_index]

    X_test = X[test_index]
    y_test = y[test_index]
```
from sklearn.preprocessing import StandardScaler
from sklearn import metrics as mt
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline

# Normalize the best training set and the test set. Normalizing transforms all of the data to the same scale

# scale attributes by the training set
scl_obj = StandardScaler()
scl_obj.fit(X_train)  # find the scalings for each column that make a zero mean and unit stdv

X_train_scaled = scl_obj.transform(X_train)  # apply to the training set
X_test_scaled = scl_obj.transform(X_test)  # apply those means and std to the test set

# train the model
lr_clf = LogisticRegression(penalty='l2', class_weight='balanced', C=1, n_jobs=-1)  # lower C value for stronger regularization since
# the data is normalized
# n_jobs is set to -1 to utilize all cores
# class_weight is balanced to help prevent the model for getting a 90% accuracy saying everything is not raining
# Balanced class weight means the number of samples divided by the number of classes(2) times the number of occurrences
lr_clf.fit(X_train_scaled, y_train)

y_hat = lr_clf.predict(X_test_scaled)

lrAcc = mt.accuracy_score(y_test, y_hat)
lrConf = mt.confusion_matrix(y_test, y_hat)
View model accuracy

```python
code here
```

```
print 'Model accuracy: %s' % lrAcc

Model accuracy: 0.735576923077

In [42]: mt.roc_auc_score(y_test, y_hat)

Out[42]: 0.57865497076023387
```
Examine model weights
Test the model by manually passing in observations

```python
In [49]:
# Let's test the model, assuming the temperature is 70 and the humidity is 80.
x = np.array([70,90])
x_reshaped = x.reshape([-1,1])
pipeline.predict(x_reshaped)

/Users/andrewclark/anaconda2/lib/python2.7/site-packages/sklearn/utils/validation
warnings.warn(msg, _DataConversionWarning)

Out[49]: array([[1]])
```
Conclusion and Recap

- What machine learning is.
- Why machine learning is important.
- Why we need machine learning audits.
- What constitutes a machine learning audit.
- What a machine learning audit entails.
- Overview of the CRISP-DMA framework.
- Simple end to end machine learning audit example using the CRISP-DMA framework.
Thank you!

- Email: andrewtaylorclark@gmail.com
- GitHub: aclarkData
- Blog: https://aclarkdata.github.io/
- LinkedIn: www.linkedin.com/in/andrew-clark-b326b767