Protecting Your Machine Learning Against Drift: An Introduction
Oliver Cobb - Applied Machine Learning Researcher
We will look at

- What drift is and why it pays detect it.
- The different types of drift.
- How drift can be detected in a principled manner.
- The anatomy of a drift detector.
- Demystify concepts such as ‘online detectors’, ‘permutation tests’, ‘MMD test’ etc.
- Practical demonstration with alibi-detect.
Preliminaries

- Wish to use $y$ for some prediction/output.
- Can't observe $y$, can observe $x=(x_1,\ldots,x_d)$ that are related.
- We fit a model $M$ to predict $y$ from $x$ and use $\hat{y} = M(x)$ as the prediction/output.

- Performance on held out data gives estimate for future performance… *Assuming the process underlying $x$ and $y$ remains constant.*
What is drift?

- When the process underlying \( x \) and \( y \) during deployment differs from the process that generated the training data.
- Can no longer expect the model's performance during deployment to match that observed on held out training data.

\[ p(x,y) \]

What is drift?

\[ p(x, y) = p(y|x)p(x) = p(x|y)p(y) \]
What is drift?

Labels available: **Supervised** drift detection

-> Can monitor model performance directly.
What is drift?

Labels unavailable: **Unsupervised** drift detection

-> Can’t monitor model performance. Must operate in high dimensions.

- **No Drift**

- **Concept Drift**
  \[ \Delta p(y|x) \]

- **Prior Drift**
  \[ \Delta p(y) \]

- **Covariate Drift**
  \[ \Delta p(x) \]
Change or chance?

- We don't expect new data to look identical to training data.
- So how do we differentiate systemic change from natural fluctuations?
- Statistical hypothesis testing!
Statistical Hypothesis Testing

- Before observing data $Z$, specify null and alternative hypothesis, $H_0$ and $H_1$.
- Specify test statistic $S(Z)$ we expect to be small if $H_0$ and large if $H_1$.
- Observe data, compute $S(Z)$, compute $\hat{p} = P(\text{such an extreme } S(Z) \mid H_0)$.
- Low $p$-value discredits $H_0$.
  - Typically specify a threshold $p$ (FPR) in advance and reject null if $\hat{p} < p$. 

---

![Scatter plots](attachment:image.png)
Offline Drift detection

- Let \( q_0 \) be distribution underlying training data, \( Z_0 \).
- Let \( q_1 \) be the distribution underlying a batch of new data, \( Z_1 \).
- \( H_0: q_0 = q_1 \). \( H_1: q_0 \neq q_1 \).
- \( S(Z_0, Z_1) \) small if \( H_0 \) true, large if \( H_1 \) true.
- Compute \( \hat{p} \). Flag drift if \( \hat{p} < p \) for desired FPR \( p \).

The hard part!
Online Drift detection

- Data points $z=(x,y)$ arrive in sequence, $z_1,z_2,...$ and we'd like to detect drift ASAP.
- Assumption:
  
  $$z_i \sim \begin{cases} q_0 & \text{for } i < T^* \\ q_1 & \text{for } i \geq T^* \end{cases}$$

- At each time $t$ we perform a hypothesis test of $\{H_0: T^* > t\}$ vs $\{H_1: T^* \leq t\}$.
  
  i.e. “has drift occurred yet”?
Online Drift detectors - desired properties

. When a change occurs the detector is fast to respond.
  ◦ i.e. Expected Detection Delay, $\text{EDD} = E[T' - T^*]$, is small.

. Ability to specify the frequency of false detections in the absence of change.
  ◦ i.e. We can specify Expected Run Time, $\text{ERT} = E[T' | T^* = \infty]$.

. There's an ERT vs EDD tradeoff.

Often overlooked
Windowing Strategies

- How do we apply SHT to data arriving sequentially?
- By collecting instances into “test” windows.
- These can then compared to the fixed “reference” window.
- Windows can be fixed sized and disjoint, fixed size and overlapping or adaptive.

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>x_1</td>
<td>0.21</td>
<td>1.51</td>
<td>-1.03</td>
<td>-0.08</td>
<td>1.46</td>
<td>0.80</td>
<td>1.13</td>
<td>0.17</td>
<td>-0.35</td>
<td>-0.12</td>
</tr>
<tr>
<td>x_2</td>
<td>-0.41</td>
<td>0.05</td>
<td>0.59</td>
<td>-0.94</td>
<td>0.73</td>
<td>0.74</td>
<td>-1.52</td>
<td>0.66</td>
<td>-1.19</td>
<td>-0.72</td>
</tr>
</tbody>
</table>

Disjoint

TEST!

TEST!
Windowing Strategies

- How do we apply SHT to data arriving sequentially?
- By collecting instances into “test” windows.
- These can then compared to the fixed “reference” window.
- Windows can be fixed sized and disjoint, fixed size and overlapping or adaptive.

### Overlapping

<table>
<thead>
<tr>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>0.21</td>
<td>1.51</td>
<td>-1.03</td>
<td>-0.08</td>
<td>1.46</td>
<td>0.80</td>
<td>1.13</td>
<td>0.17</td>
<td>-0.35</td>
<td>-0.12</td>
</tr>
<tr>
<td>(x_2)</td>
<td>-0.41</td>
<td>0.05</td>
<td>0.59</td>
<td>-0.94</td>
<td>0.73</td>
<td>0.74</td>
<td>-1.52</td>
<td>0.66</td>
<td>-1.19</td>
<td>-0.72</td>
</tr>
</tbody>
</table>
Windowing Strategies

- How do we apply SHT to data arriving sequentially?
- By collecting instances into “test” windows.
- These can then compared to the fixed “reference” window.
- Windows can be fixed sized and disjoint, fixed size and overlapping or adaptive.

### Adaptive

<table>
<thead>
<tr>
<th>t</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>0.21</td>
<td>1.51</td>
<td>-1.03</td>
<td>-0.08</td>
<td>1.46</td>
<td>0.80</td>
<td>1.13</td>
<td>0.17</td>
<td>-0.35</td>
<td>-0.12</td>
</tr>
<tr>
<td>$x_2$</td>
<td>-0.41</td>
<td>0.05</td>
<td>0.59</td>
<td>-0.94</td>
<td>0.73</td>
<td>0.74</td>
<td>-1.52</td>
<td>0.66</td>
<td>-1.19</td>
<td>-0.72</td>
</tr>
</tbody>
</table>

**TEST!**

**KEEP?**
Different windowing strategies call for different test statistics and threshold determination processes.
They also each have their own pros and cons.
Disjoint Window Detectors

- Tests are independent and performed infrequently.

- Can compute p-value corresponding to any test statistic \( S(Z_0, Z_1) \)!
  
  **Achieved via a permutation test:**
  
  - \( \text{shuffle}(Z_0, Z_1) : (Z_0, Z_1) \mapsto (Z^*_0, Z^*_1) \)
  
  - \( \text{alt_stats} = [S(\text{shuffle}(Z_0, Z_1)) \text{ for } _\_ \text{ in range(B)}] \)
  
  - \( \hat{p} = (\text{alt_stats} > S(Z_0, Z_1)).\text{mean()} \)

- Can learn new test statistic \( S \) for each test.

- Sensitive to choice of window size.

- Slow to respond to severe drift.
Test statistics are correlated.
- Makes controlling ERT very difficult.

- Test statistic must be:
  - Incremental
  - Pre-specified

- Computationally light.
- Fast to respond to severe drift.
Adaptive Window Detectors

- Typically accumulate some notion out ‘outlierness’.
- Hard to control ERT.

- Adaptive window size.
- Accumulating ‘outlierness’ not good for EDD!
- Why not?

Image credit: Basseville and Nikiforov (1993) - Detection of Abrupt Changes: Theory and Application
Drift = persistent outliers?

$q_0 = \mathcal{N}(0, I_2)$
Drift = persistent outliers?

$q_0 = N(0, I_2)$

$q_1 = N(\mu, \sigma I_2)$
Drift = persistent outliers?

\[ q_0 = N(0, I_2) \]

\[ q_1 = N(\mu, \sigma I_2) \]
Drift = persistent outliers?

$q_0 = N(0, l_2)$

$q_1 = N(\mu, \sigma l_2)$
Drift = persistent outliers?

$q_0 = N(0, I_2)$

$q_1 = N(\mu, \sigma I_2)$
Test Statistics

- Outlierness-based test statistics are not sufficient - so what is?
- Test statistics which estimate distance between $q_0$ and $q_1$.
- Could first estimate $\hat{q}_0$ and $\hat{q}_1$ and evaluate $d(\hat{q}_0,\hat{q}_1)$.
- More efficient to directly estimate $d(q_0,q_1)$.
- $d(q_0,q_1) = \text{MMD}_k(q_0,q_1)$ features prominently in alibi-detect.
Maximum Mean Discrepancy (MMDₖ)

- Transforms problem from specifying a distance \( d(q₀,q₁) \) between distributions to specifying a similarity (kernel) \( k(z₀,z₁) \) between data points.
- Typically \( k(z₀,z₁) = \Phi(z₀)^T\Phi(z₁) \) for some projection \( \Phi \).

\[
\text{MMD}_k(q₀,q₁) = \mathbb{E}[k(z₀,z₀') + k(z₁,z₁') - 2k(z₀,z₁)]
\]

\[
\text{MMD}_k(q₀,q₁) = \text{Avg. similarity between reference instances} \\
+ \text{Avg. similarity between test instances} \\
- 2*\text{Avg. similarity between reference and test instances}
\]

\[
\frac{1}{nₓ(nₓ - 1)} \sum_{x₁ ≠ x₂ ∈ ℱ} k(x₁, x₂) + \frac{1}{nᵧ(nᵧ - 1)} \sum_{y₁ ≠ y₂ ∈ ℳ} k(y₁, y₂) - \frac{2}{nₓnᵧ} \sum_{(x,y) ∈ ℱ × ℳ} k(x, y)
\]
Maximum Mean Discrepancy (MMD$_k$)

\[
\text{MMD}_k(q_0, q_1) = \text{Avg. similarity between reference instances (small)} \\
+ \text{Avg. similarity between test instances (large)} \\
- 2\times \text{Avg. similarity between reference and test instances (small)} \\
= \text{large (e.g. 0.1 + 0.8 - 2\times 0.12 = 0.68)}
\]
Maximum Mean Discrepancy (MMD\(_k\)) - Permuted

\[ MMD_k(q_0,q_1) = \text{Avg. similarity between reference instances (small)} + \text{Avg. similarity between test instances (small)} - 2*\text{Avg. similarity between reference and test instances (small)} \]

= small (e.g. 0.09 + 0.11 - 2*0.09 = 0.02)
### Summary and demo

**ALIBI DETECT**

**Drift Detection**

<table>
<thead>
<tr>
<th>Detector</th>
<th>Tabular</th>
<th>Image</th>
<th>Time Series</th>
<th>Text</th>
<th>Categorical Features</th>
<th>Online</th>
<th>Feature Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kolmogorov-Smirnov</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Maximum Mean Discrepancy</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Least-Squares Density Difference</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Chi-Squared</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mixed-type tabular data</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Classifier</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Classifier Uncertainty</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Regressor Uncertainty</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

[github.com/SeldonIO/alibi-detect](https://github.com/SeldonIO/alibi-detect)
Thanks for watching!

github.com/SeldonIO/alibi-detect

@SeldonResearch

oc@seldon.io