Machine Learning Monitoring Concepts
by Hakim Elakhrass (Co-founder of NannyML) and datacamp

Reasons for models to fail
-software issues (bugs)
-drifts in the input data
-changes in relationship between features and targets

Improving AI safety
-bias
-adversarial attacks - detect malicious input of input data
-lack of explainability - observing the behavior of the model and its input data over time can foster model understanding and explainability

Traditional monitoring workflow alert based on drifts in input data (distributions)
-the problem is many false alerts

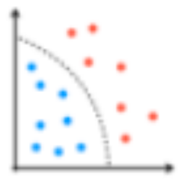
Ideal workflow
-focuses on root cause analysis (RCA), ie detects a drift and then assesses it against performance

Monitoring performance

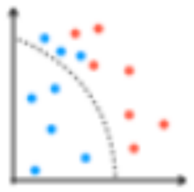
1. calculate performance when possible (ie accuracy)
2. when it is not possible to directly calculate performance then we need to estimate perfomance (ie regression or confidence scores)
3. measure business impact (key performance indicators (KPI))

*any of these are off then something is wrong with our model

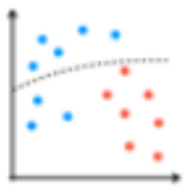
Goal of RCA
investigate:
-covariate shift - shifts in the input data distribution
-concept drift - changes in relationship between features and targets
nice visual explaining the difference:



The original model



Covariate shift



Concept drift

Issue resolution

1. retraining (most popular) - problem requires additional labeled data and additional compute time
2. refactor - taking a step back - are we using the right features, are they engineered appropriately, do we need a new type of model?
3. changing the downstream process - if the model isn't robust enough, modify processes around the prediction

Model fails to make predictions
-language barriers > combining different programming languages
-code maintenance > compatibility in original code as updates occur
-scaling > infrastructure not robust

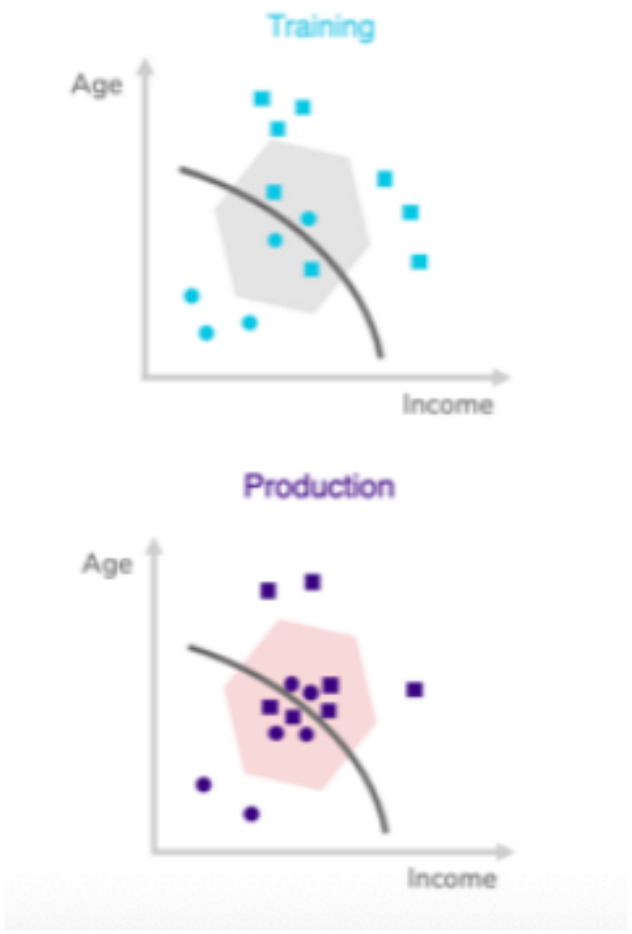
Model's performance degrades
-may be hard to diagnose, there may be no obvious alerts
meaning the pipeline or application may still be functioning but the predictions are no longer valid

1. covariate shift > change in the input’s distribution
    1. can detect with statistical tests focused on distance methods (Jenson-Shanon and Wasserstain, Komogorov-Smirnov, and Chi-squared)
    2. *remember not every drift impacts performance
2. concept drift > change in the relationship between the input data and the targets
    1. difficult to detect
    2. *almost always affects the business impact of the model

Monitoring technical performance directly
a covariate is just another name for an input feature
Three covariate shifts:

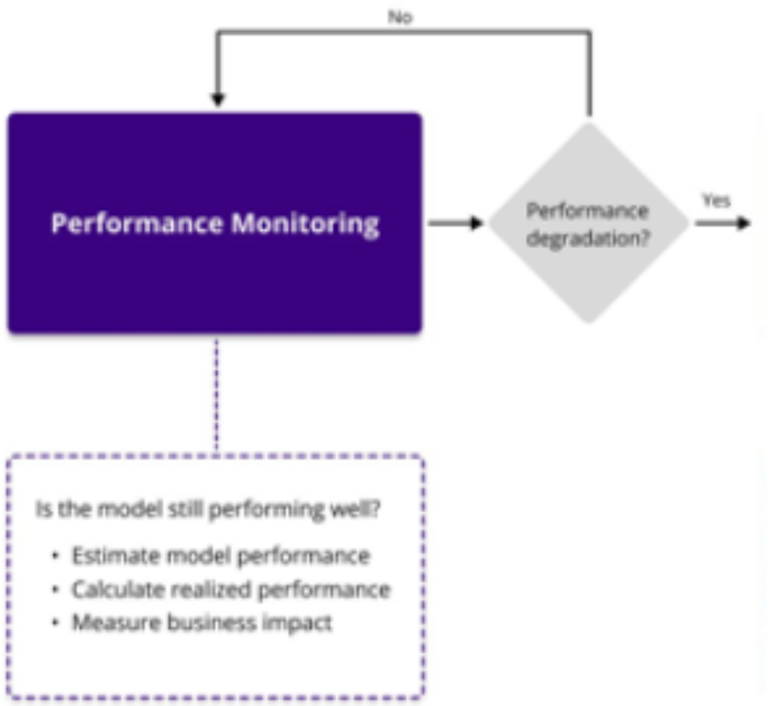
1. data shifting to regions where the model is more certain of the impact > no impact or possible positive impact
2. data shifting to regions with more production data from under-represented segments in the training set > unknown impact
3. data shifting to regions where the model is less certain (close to the decision boundary the the model tried to learn) > negative impact

Guaranteed negative impact
features shift to uncertain regions closer to the decision boundary > ML model’s performance will decrease > always negative impact
Visualize:



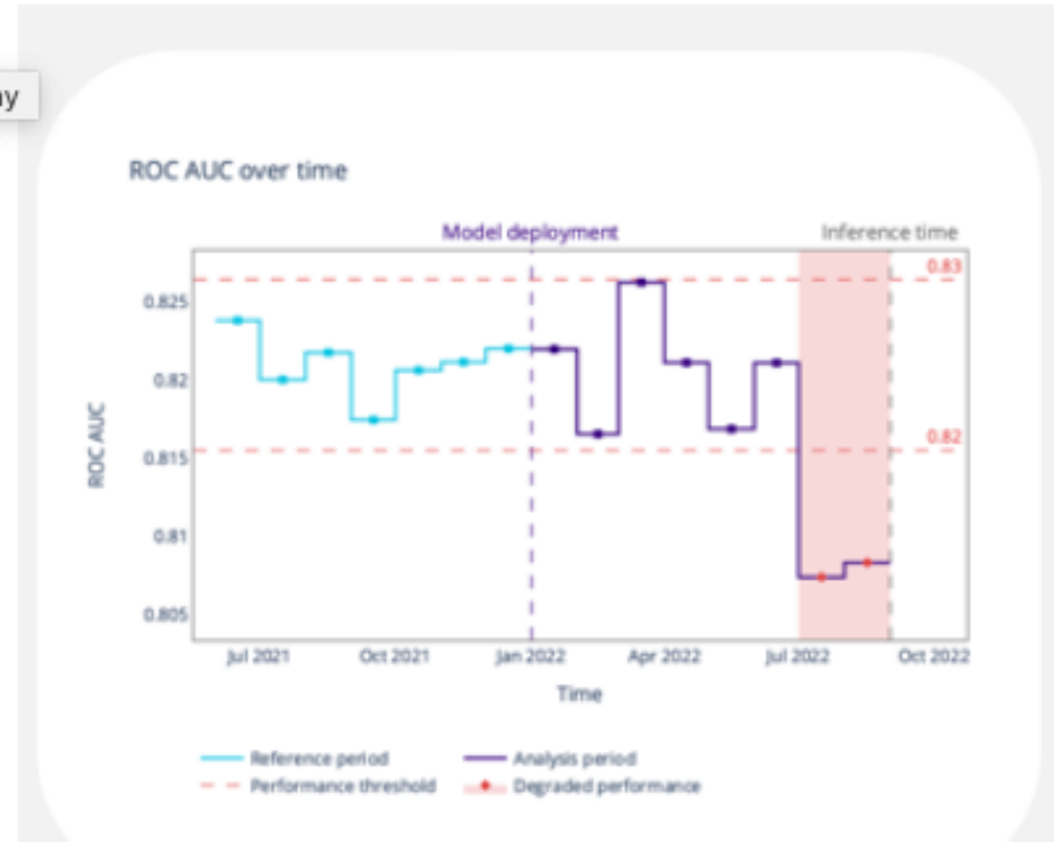
False alerts problem
covariate shifts to unseen regions can have a negative impact, but it does not appear often
drift detection methods don’t have a bulit-in logic to distinguish the type of shift >
*assume every shift will affect model performance
*features can shift but not affect the model’s performance, if they are irrelevant
these systems can do more harm than good

Technical performance of an ML model is a direct metric of how well the model performs the task at hand
*first step of the monitoring workflow in production

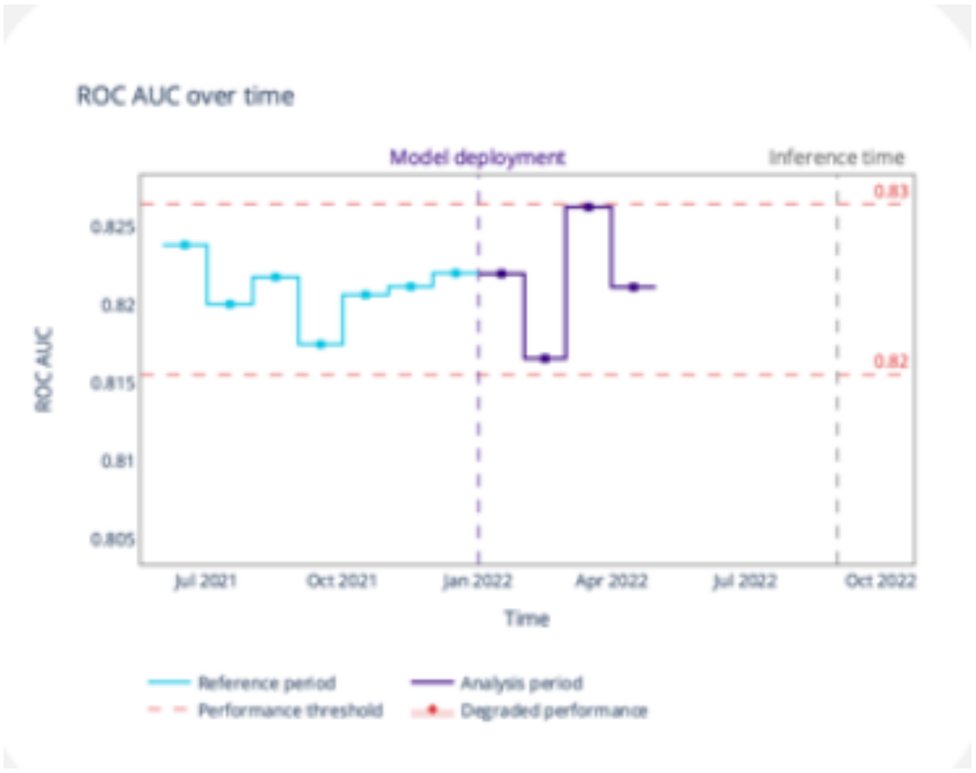

No
Performance Monitoring
Performance degradation?
Yes
Is the model still performing well?
• Estimate model performance
• Calculate realized performance
• Measure business impact

Availability of ground truth
assessing taxi arrival estimation (example of instant ground truth)
visual showing reference to deployment and analysis

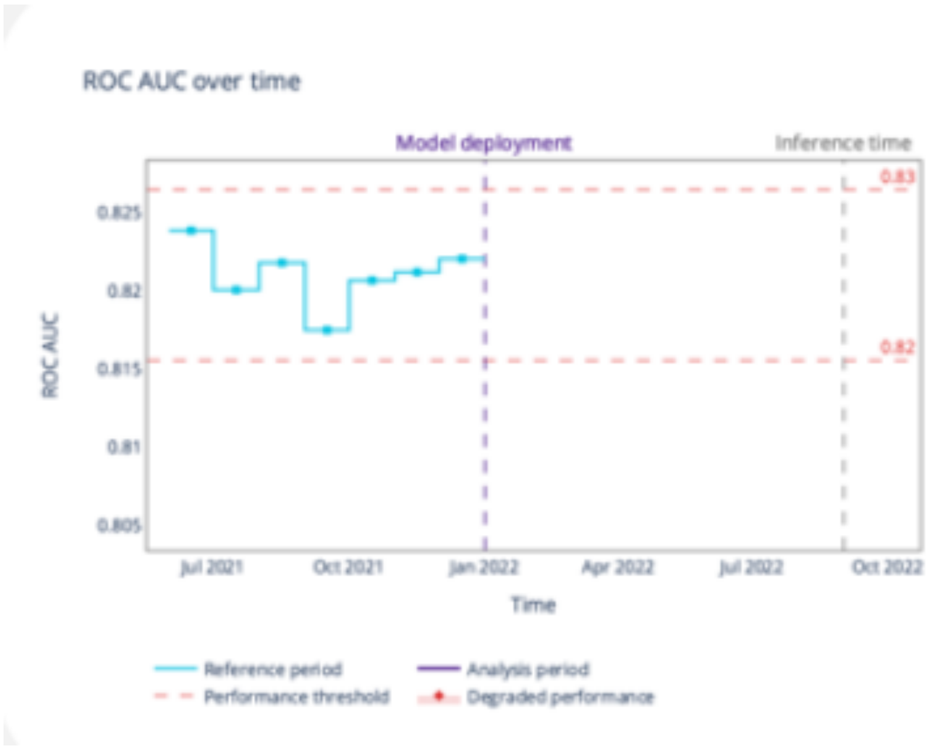
ay



visual of loan churning (example of delayed ground truth)



Absent ground truth (example insurance pricing)



*here as you can see it becomes very complicated to determine how well our model is performing

Performance estimation
two algorithms:

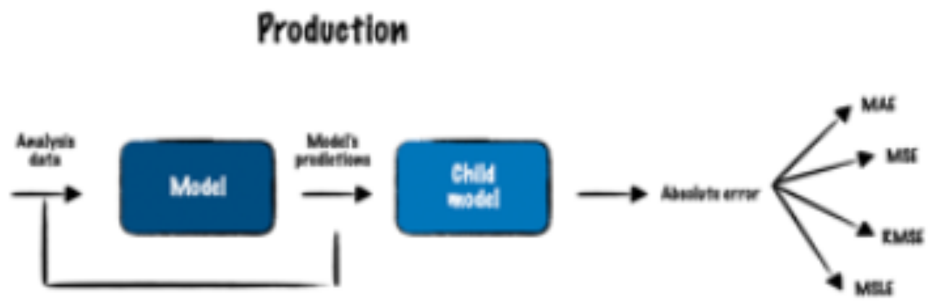
- CBPE (Confidence-Based Performance Estimation) for classification tasks
- DLE (Direct Loss Estimation) for regression tasks

CBPE used for classification tasks that utilizes the confidence score of a model's predictions to estimate the confusion matrix
model predicts that instance is a percent correct and a percent incorrect (positive or negative)
process is repeated for all examples
aggregated confidence scores generate an estimated confusion matrix
we can then calculate accuracy, precision, recall, F1-score
*if the model is negatively affected by the covaritate shift, the performance estimation will capture this impact

CBPE is not perfect
assumptions need to be made
first need to assume no covariate shift in the unseen regions
exampel loan default model trained on 40-70yo dropped into a 40 or below market
> these estimations can not be expected to be reliable

second there can be no concept drift present in the incoming data
concept drift refers to changes in the relationship between input features and targets
this can cause the model’s decision boundary to become outdated > making its predictions no longer valid
thirdly probability calibration is required
*by default different ML models are not calibrated
they can be calibrated before being put into production

DLE (direct loss estimation) is a technique involving the prediction of the absolute error of the model for regression tasks
this error represents the uncertainty associated with the model’s output
DLE achieves this using an external “child model” > this is a popular ML algorithm called LightGBM
LightGBM trained on reference data and the main model’s prediction
LightGBM allows for the calculation of various regression error metrics (ie MAE, MSLE)
DLE captures the presence of a covariate shift in the input data



DLE aslo assumes no covariate shift in the unseen regions and no concept drift is present in the incoming data
good to consider that DLE does add additional complexity to the system by employing another model to estimate the performance
this can lead to increased computational resources

What is covariate shift?
again covairate variables = input features
the distribution of covariates (noted as P(X)) changes, while the conditional probability of the output given the input (noted as P(Y|X) remains unchanged
accurate definition of covariate shift is the changes in the joint distribution of the covariates

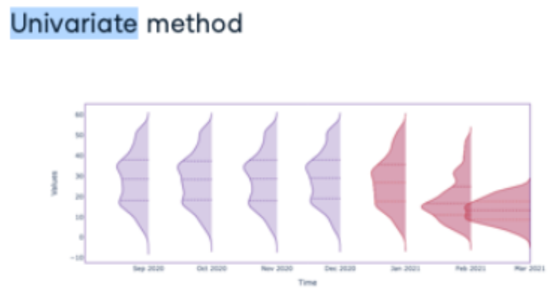
Why joint probability distribution?
there are instances of covariate shift where if you examine each feature

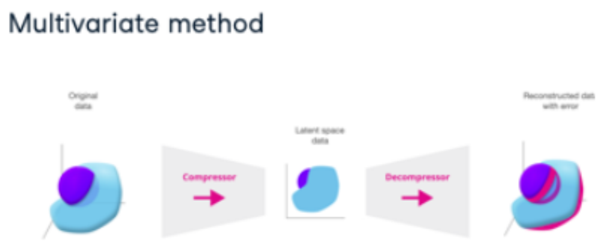
separately, you won’t notice a change in distribution
example is accidentally swapping two features > the distributions would appear almost the same
as joint we would see correlation shifts from positive to negative

Why does covariate shift occur?
- real world is dynamic, patterns and trends change
- variations in how data is collected between testing and production
- model trained on a certain version of software then later applied on an updated version or on features with evolving behaviors

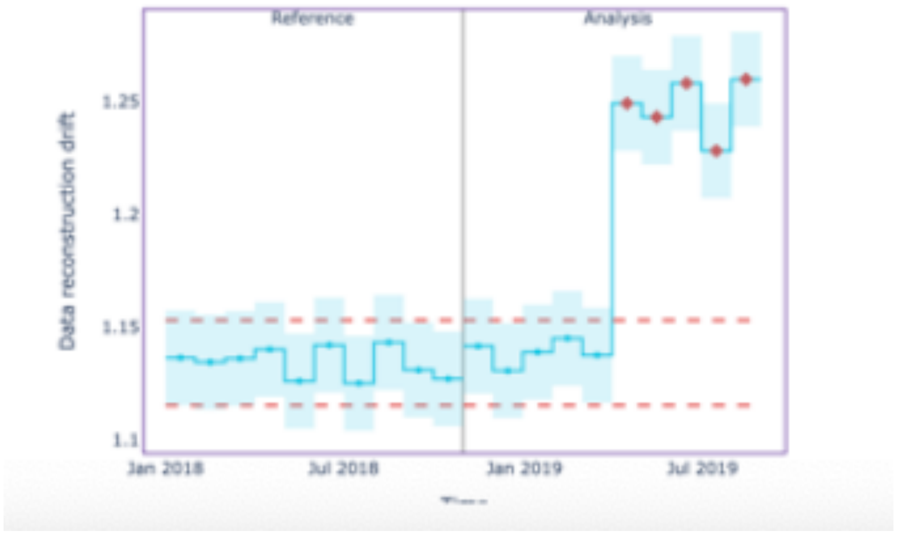
How does covariate shift occur?
-sudden, gradual, or seasonal

How to detect the covariate shift?





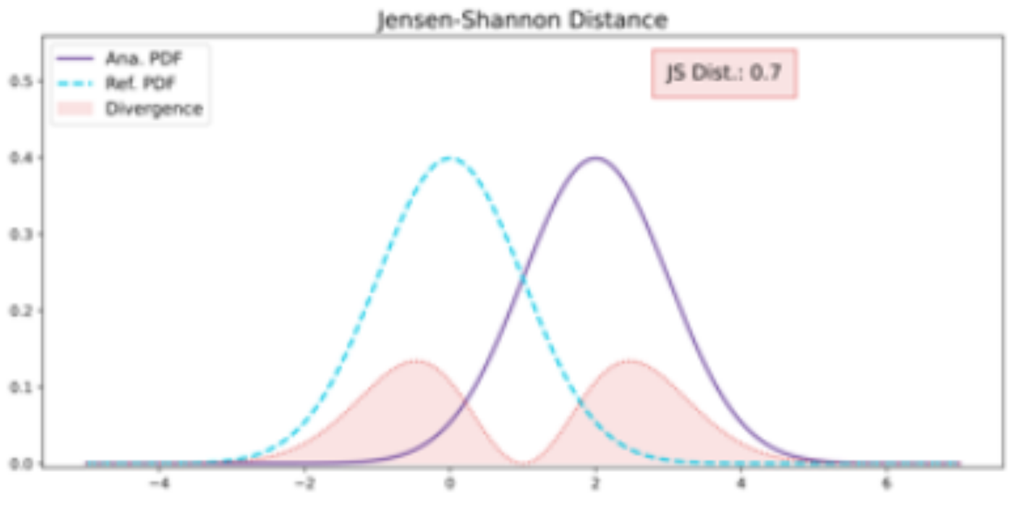
Multivariate drift detection
looks for changes in joint distribution
uses the PCA algorithm
which compresses the data into a lower dimension, aiming to capture the internal structure of the model input data while filtering out random noise
then the PCA algorithm utilizes inverse PCA to reonstruct the data back to its original shape with a certain level of error
then we use reconstructioin error as a measure of drift
visual:



by comparing the reconstruction error to a baseline without shift, we can determine whether there has been a change in the input data distribution

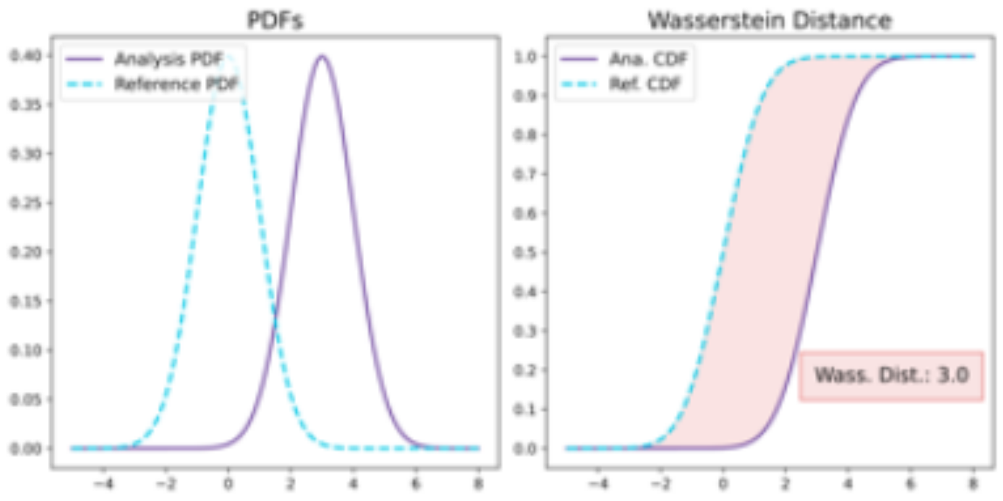
Once we verify the occurrence of a shift in the incoming data > we need to next pinpoint the single features that are undergoing the drift
here we use different methods depending on the type of variable (continuous vs categorical)

Continuous methods (Jensen-Shannon)
measures the similarity of two distributions using Kullback-Leibler divergence
operates in the range of 0 to 1
is sensitive to small drifts

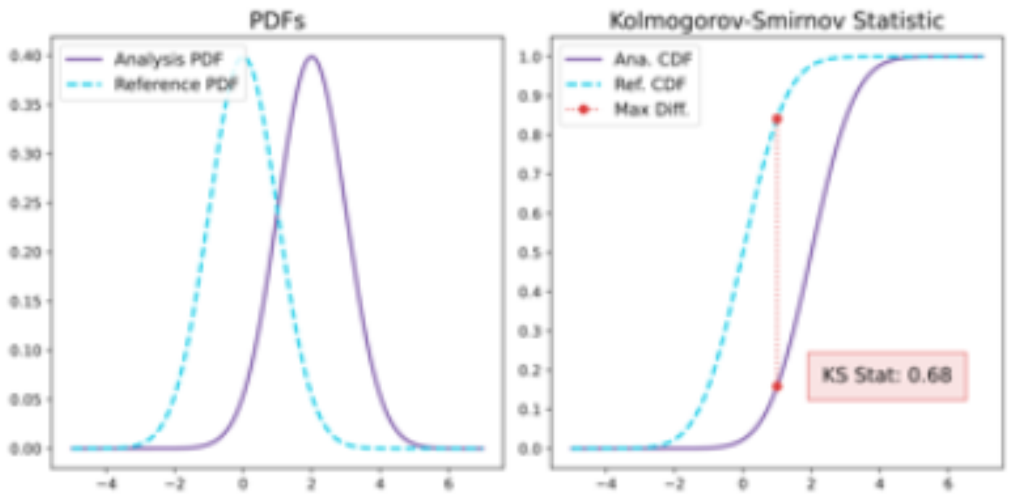


Continuous methods (Wasserstein distance)
quantifies the minimum effort needed to transform one distribution into another

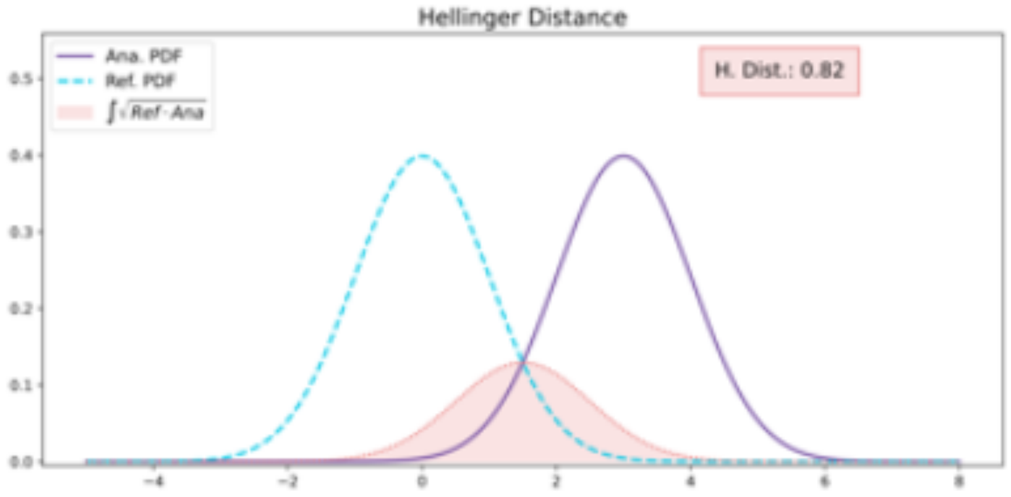
metric ranges from 0 to infinity
*be wary of outliers, can significantly impact the results > this method is less robust to outliers



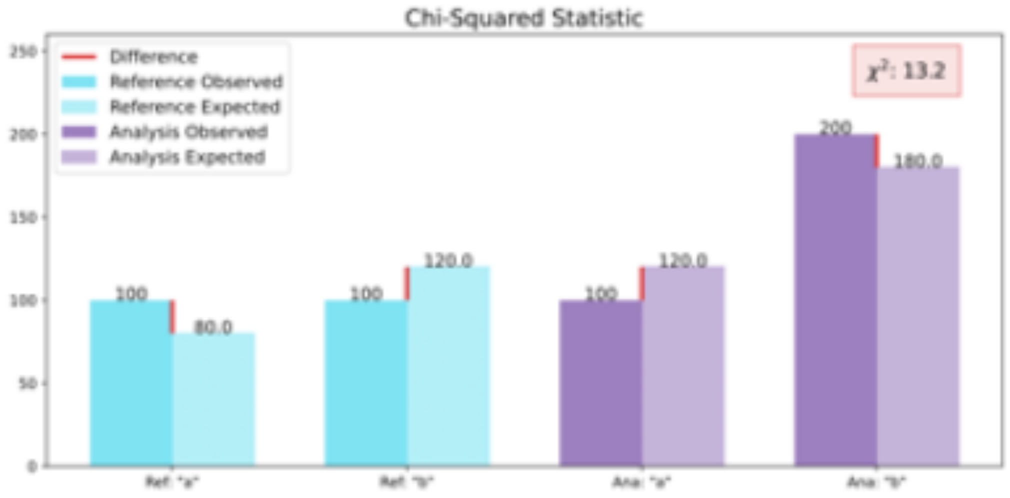
Continous methods (Kolmogorov-Smirnov statistic test)
is the maximum distance of the cumulative distribution functions of the two samples
falls into 0-1 range
limited with larger datasets > may generate false positive alerts for drifts, increasing the chances of misidentifying meaningful changes



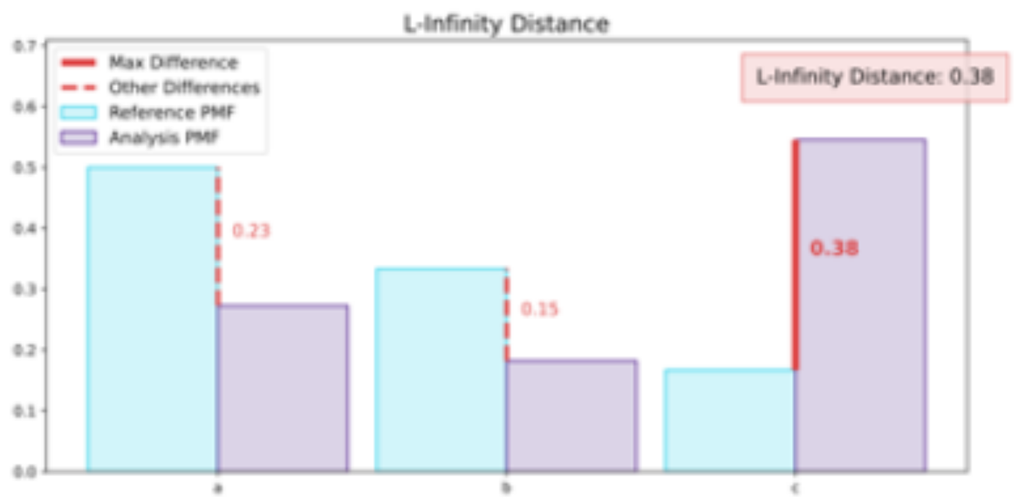
Continuous and categorical methods (Hellinger method)
measures the overlap between distributions
*it can’t detect shifts when there’s no overlap > means that even if distributions are close or far apart, it still results in maximum value



Categorical methods (Chi-squared test)
sensitive to changes in low-frequency categories
a small change can significantly impact the test statistic when the frequency is already low



Categorical methods (L-infinity)
measures the largest difference between distributions of different categories
works well with numerous categories as it identifies the most significant shift across all categories > effectively detecting differences regardless of the number of categories
recommended if your specific aim is to detect changes in individual categories



*can also use Jensen-Shannon when dealing with many categories

What is concept drift?
a change in the relationship between the model inputs and the targets
training P(Y|X) is not equal to the production P(Y|X) and P(X) stays the same

Why concept drift happens?
1. external events ie policy changes
2. unmodeled seasonality ie Black Friday
3. changes in data-generation process ie interface change
4. evolving user behavior ie habits change with a system

The relationship between features and targets is referred to as a “concept”
The dynamics of concept drift are similar to covariate drift > sudden, gradual, reoccurring (ie black Friday)
Covariate and concept drift can appear together or separately

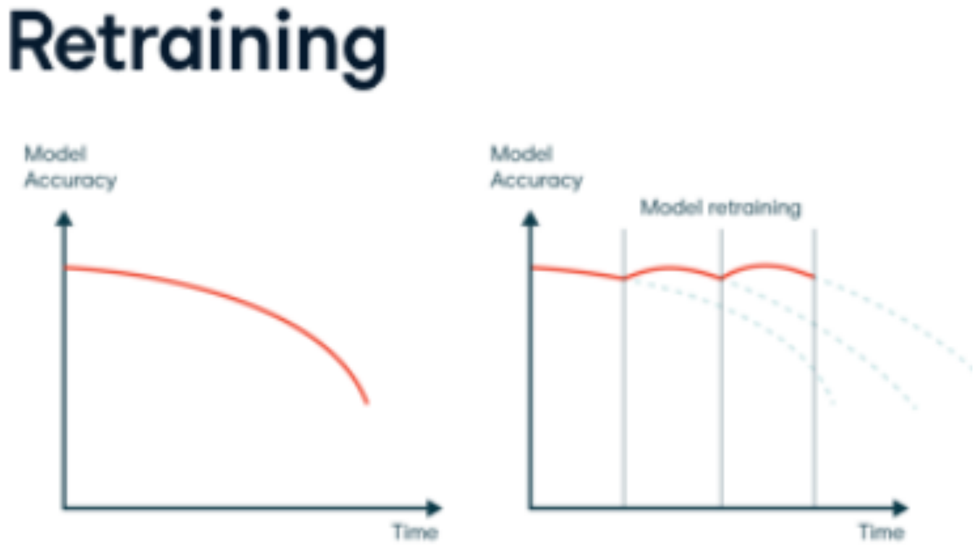
Effects of covariate shift on concept drift
- negative > the effect of concept drift decreases
- positive > the effect of concept drift intensifies

Concept drift detection
-error-based methods > tracking error changes over time (this requires ground truth)
-train a new model using training and production data > *changein the predictions is a concept drift (can be expensive)

How to handle concept drift?

ML model in its nature is static and doesn’t adapt to the changes in the environment

Solution 1 - Retraining
periodic or trigger-based retraining can keep the mode up to date with recent patterns
downsides to retraining are the more frequently you update your model, the more opportunities there are for updates to fail
high computing resources
does not guarantee a solution
need to keep look out for problems in the downstream processes, data leakage, and training-serving skew
visual:



Solution 2 - Online learning
also known as incremental or streaming learning
models are trained and updated continuously as new data arrives
benefits include ability to handle evolving data streams and adapt in real-time to changing conditions
can capture concept drift and provide timely insights
computationally efficient as it processes data instances one at a time > making it suitable for large-scale/high-velocity data scenarios
limitations > requires constant access to ground truth, can be sensitive to noise or erroneous data, likely requires careful parameter tuning to maintain model performance over time