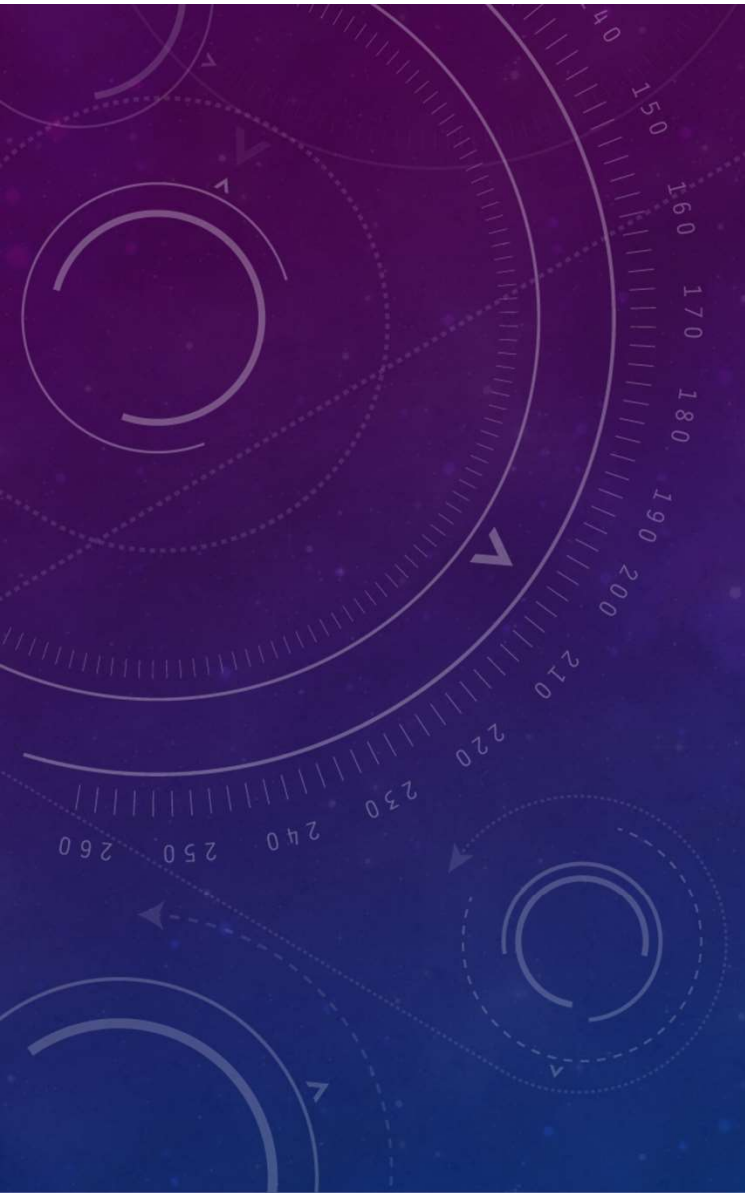


MACHINE LEARNING WITH .NET

Asmaliza Ahzan @ Emma

asmaliza@iverson.com.my

ANOMALY DETECTION MODEL

CHAPTER 6

ANOMALY DETECTION

- Anomaly detection is one of the newer additions to ML.NET, and specifically, time-series transforms.
- Without needing to do manual spot-checking, anomaly detection algorithms train on this data and determine whether there are any anomalies.
- This task creates an anomaly detection model by using Principal Component Analysis (PCA). PCA-Based Anomaly Detection helps you build a model in scenarios where it is easy to obtain training data from one class, such as valid transactions, but difficult to obtain sufficient samples of the targeted anomalies.

PCA-BASED ANOMALY DETECTION

- An established technique in machine learning, PCA is frequently used in exploratory data analysis because it reveals the inner structure of the data and explains the variance in the data.
- PCA works by analyzing data that contains multiple variables.
- It looks for correlations among the variables and determines the combination of values that best captures differences in outcomes.
- These combined feature values are used to create a more compact feature space called the principal components.

EXAMPLE SCENARIOS

- Identifying transactions that are potentially fraudulent.
- Learning patterns that indicate that a network intrusion has occurred.
- Finding abnormal clusters of patients.
- Checking values entered into a system.

DIVING INTO THE RANDOMIZED PCA TRAINER

- The randomized PCA trainer is the only traditional trainer for anomaly detection found in ML.NET currently.
- The randomized PCA trainer requires normalization of the values; however, caching is not necessary and no additional NuGet packages are required to utilize the trainer.
- The randomized PCA trainer requires normalization of the values; however, caching is not necessary and no additional NuGet packages are required to utilize the trainer.
- The algorithm finds edge cases if the computed error is not close to 0. If it finds the error is close to 0, it is considered a normal data point (that is, a non-anomaly).

DIVING INTO TIME SERIES TRANSFORMS

- Time series support was added as a series of transforms to be applied to your training and test data.
- Time series, as mentioned previously, is also one of the newer additions to ML.NET, being added in 1.2.0.

TIMESERIESCATALOG CLASS

Class	Descriptions
DetectAnomalyBySrCnn	Detects anomalies with the SRCNN algorithm
DetectChangePointBySsa	Detects anomalies with the Singular Spectrum Analysis (SSA) algorithm on change points
DetectIidChangePoint	Detects changes to predict change points with an independent identically distributed (i.i.d) algorithm
DetectIidSpike	Detects changes with an i.i.d algorithm but predicts spikes instead of change points
DetectSpikeBySsa	Detects spikes using the SSA algorithm
ForecastBySsa	Uses the SSA algorithm for a singular variable- (commonly referred to as univariate-) based time series forecasting

CREATING A TIME SERIES APPLICATION

- The application we will be creating is a network traffic anomaly detector.
- Given a set of attributes relating to the network traffic amount (in bytes), the application will use that data to find anomalies in the amount of traffic for a given checkpoint.
- As with other applications, this is not meant to power the next ML network traffic anomaly detection product; however, it will show you how to use time series in ML.NET, specifically to detect spikes with SSA.

PREDICTOR OUTPUT

- The output includes the three data points: HOST, TIMESTAMP, and TRANSFER.
- The new additions are ALERT, SCORE, and P-VALUE.
- ALERT values of nonzero indicate an anomaly.
- SCORE is a numeric representation of the anomaly score; a higher value indicates a spike.
- PVALUE, a value between 0 and 1, is the distance between the current point and the average point. A value closer or equal to 0 is another indication of a spike.

CREATING AN ANOMALY DETECTION APPLICATION

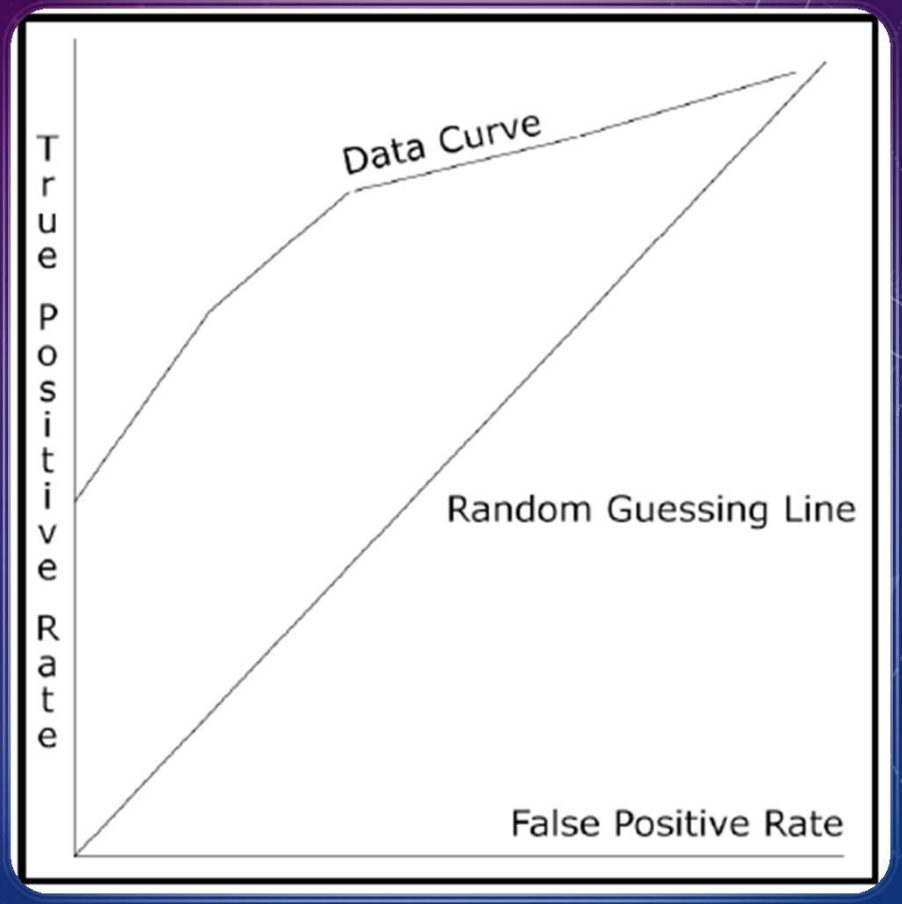
- The application we will be creating is a login anomaly detector.
- Given a set of attributes relating to the login, the application will use that data to find anomalies such as unusual login times.

AREA UNDER THE ROC CURVE

- This computed area is equal to the chance that the algorithm, randomized PCA, in our case, scores a positive instance higher than a negative one, both chosen randomly to better evaluate the data.
- The number returned closer to 100% is the ideal value, while if it is closer to 0%, you will more than likely have significant false positives.
- You might remember our earlier example application getting 78%. This means that there was a 22% chance of a false positive; the following outlines some suggestions to improve the model and should reduce this number.

AREA UNDER THE ROC CURVE

- The following diagram visually reflects both a random guessing line and an arbitrary data curve.
- The area under the data curve in between the random guessing line is the area under the ROC curve data metric.



DETECTION RATE AT FALSE POSITIVE COUNT

- The detection rate at false positive count property is the detection rate of K false positives.
- A false positive in an anomaly detection scenario would be to consider a data point an anomaly when, in fact, it was not.
- This rate is computed as follows:

$$Detection\ Rate\ of\ K\ False\ Positives = X / Y$$

- Here, X is calculated to be the top test samples based on the scores previously described in the anomaly detection example (sorted in descending order).
- These are considered the top true positives (that is, more likely to be actual anomalies).

DETECTION RATE AT FALSE POSITIVE COUNT

- Y is calculated to be the total number of anomalies in the test data regardless of the score value (not filtering to points that look suspicious or not).
- In theory, this number could be very high if the number of false positives is high in your training data.
- As you build production models with randomized PCA, ensure your data represents as close to production as possible to avoid overfitting or underfitting to anomalies.

ACTIVITY

Detect anomalies in product sales with ML.Net

THE END