Vapnik-Chervonenkis Theorie

Die VC Generalization Bound kann dazu verwendet werden die Anzahl an Samples N zu bestimmen mit denen eine Hypothese mit einer VC Dimension d_{vc} mit einer Wahrscheinlichkeit δ mit einem maximalen Fehler ε gelernt werden kann.

Aufgabe 1: Benötigte Anzahl an Samples bestimmen

In diesem Schritt soll die Anzahl der benötigten Samples, mit denen eine Hypothese mit zwei Dimensionen mit einer Wahrscheinlichkeite von 10% mit einem maximalen Fehler von $0,05$ gelernt werden kann, bestimmt werden.

Der Anzahl der Samples ist durch folgendes Theorem gegeben: $N \geq \frac{8}{\epsilon^2} \ln \frac{4((2N)^{d_{vc}} + 1}{\delta}$

Hinweis: N wird iterativ bestimmt indem mit einem geschätzten Wert begonnen wird (hier: 1000) und so lange iteriert wird bis der Wert konvergiert

```
clearvars;
clc;

e = 0.05;
delta = 0.1;

N1 = 0;
N2 = 1000;
dvc = 2;

while (abs(N2-N1) > 0.0001)
    N1 = N2;
    N2 = (8/e^2)*log(4*(((2*N1)^dvc)+1)/delta);
end

N2
```

```
N2 = 8.9191e+04
```

Aufgabe 2: VC Bound für Datensatz bestimmen

Jetzt soll für den Datensatz `features.csv` der, durch die VC-Theorie vorhergesagte Performance Grenzwert, der mit einer Wahrscheinlichkeit von 10% nicht überschritten wird, bestimmt werden.

Der Grenzwert ist durch folgendes Theorem gegeben: $E_{out}(h) \leq E_{in}(h) + \sqrt{\frac{8}{N} \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}}$

```
clearvars;
clc;

delta = 0.1;
dvc = 18;
N = 729;

e = sqrt((8/N)*log(4*(((2*N)^dvc)+1)/delta))
```

```
e = 1.2163
```

Aufgabe 3: VC-bound mit tatsächlichem Fehler vergleichen

Jetzt soll der tatsächliche Fehler eines mittels Multipler Linearer Regression gelernten Modells auf dem Datensatz ermittelt werden. Zur Bestimmung des Fehlers soll der SMAPE (Symmetric Mean Absolute Percentage Error) verwendet werden. Dieser ergibt sich wie folgt:

$$SMAPE = \frac{100\%}{n} \sum_{t=1}^{n} \frac{|y(t) - \hat{y}(t)|}{((|y(t)| + |\hat{y}(t)|)/2)}$$

```
clearvars;
clc;

data = readtable('features.csv', 'Delimiter', ',', 'TreatAsEmpty', 'true');
timestamps = table2array(data(:, 1));
features = table2array(data(:, 2 : end));

training_targets = features(1 : 729, 9);
training_features = features(1 : 729, [1 : 8, 10 : end]);
training_timestamps = timestamps(1 : 729, :);
test_targets = features(730 : end, 9);
test_features = features(730 : end, [1 : 8, 10 : end]);
test_timestamps = timestamps(730 : end, :);

X = [ones(size(training_features, 1), 1) training_features];
p = pinv(X) * training_targets;

X = [ones(size(test_features, 1), 1) test_features];
output = X * p;


SMAPE = 100 / size(test_targets.', 1) * sum(abs(output.' - test_targets.') / ((abs(test_target
```

```
SMAPE = 13.1671
```