A1.1 a) b) i., ii.

b) iii.

Die Sigmoid Funktion kann nur positive Werte annehmen, während die tanh-Funktion punktsymmetrisch um den Ursprung ist. Zudem sind die Werte der Ableitung von tanh größer als die der Sigmoid-Funktion.

1.2.

a) Der Gradient der ReLU Funktion ist für positive x-Werte konstant 1. Das heißt es kann für x>=0 keine Sättigung auftreten und der Lernvorgang ist somit effizient.

b) Für große x-Werte konvergiert der Gradient des tangens hyperbolicus im Gegensatz zu ReLU gegen 0. Somit ist die Veränderung der Gewichte/des Bias in jeder Iteration sehr gering und das Netzwerk lernt sehr viel langsamer.

c) Seien viele ui positiv und der Gradient groß und X ein Neuron im Netz => die meisten Eingänge in X positiv. Nun wird aber der Fehler zurück propagiert und die Gewichte kleiner gewählt, jedoch so viel kleiner, dass die meisten Eingänge in X im Folgenden negativ sind. Durch diese Eingänge wird darauf im Folgenden Verlauf kein Fehler mehr zurück propagiert. Insbesondere bleibt der Output von X konstant.

Mit Leaky-ReLU ist eine Aktualisierung der Gewichte auch möglich, wenn die meisten Eingänge negativ sind => Problem gelöst.

A1.3

a) Da die ELU Funktion auf ganz |R differenzierbar ist, ist der Wert auch auf ganz |R eindeutig. Dies ist bei Leaky-ReLU nicht der Fall.

b) i. Wenn der Erwartungswert und die Standardabweichung von Epoche zu Epoche stark unterschiedlich ist, bedeutet das auch, dass die Gewichte sich stark ändern. Dies macht das Lernen ineffizient.

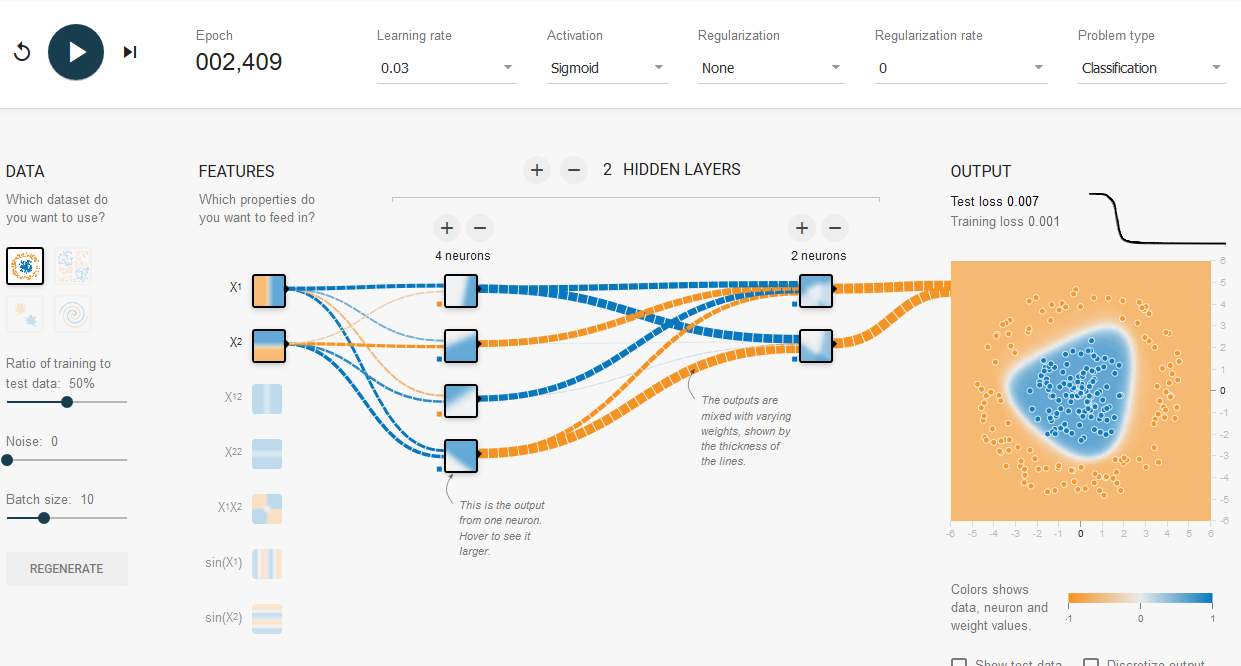


Abbildung 1: Anzahl der Epochen bei Nutzen der sigmoid-Funktion -> 2409

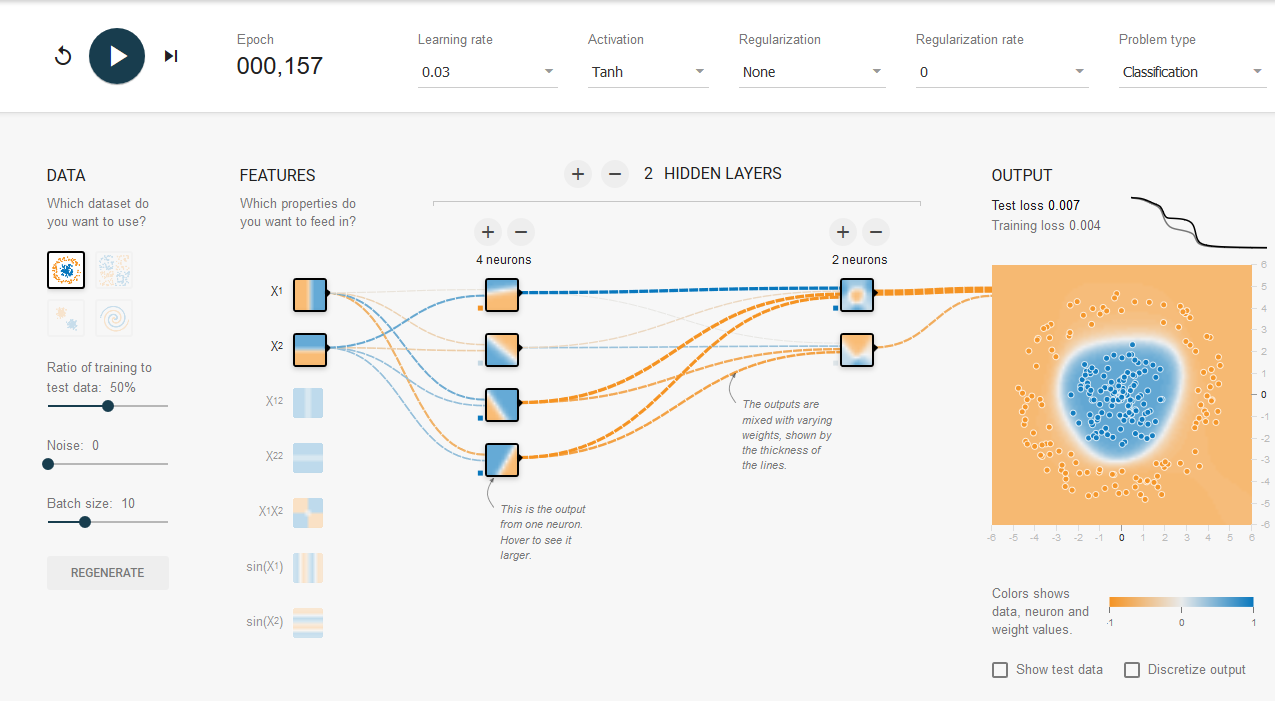


Abbildung 2: Anzahl der Epochen bei Nutzen der tanh-Funktion -> 157

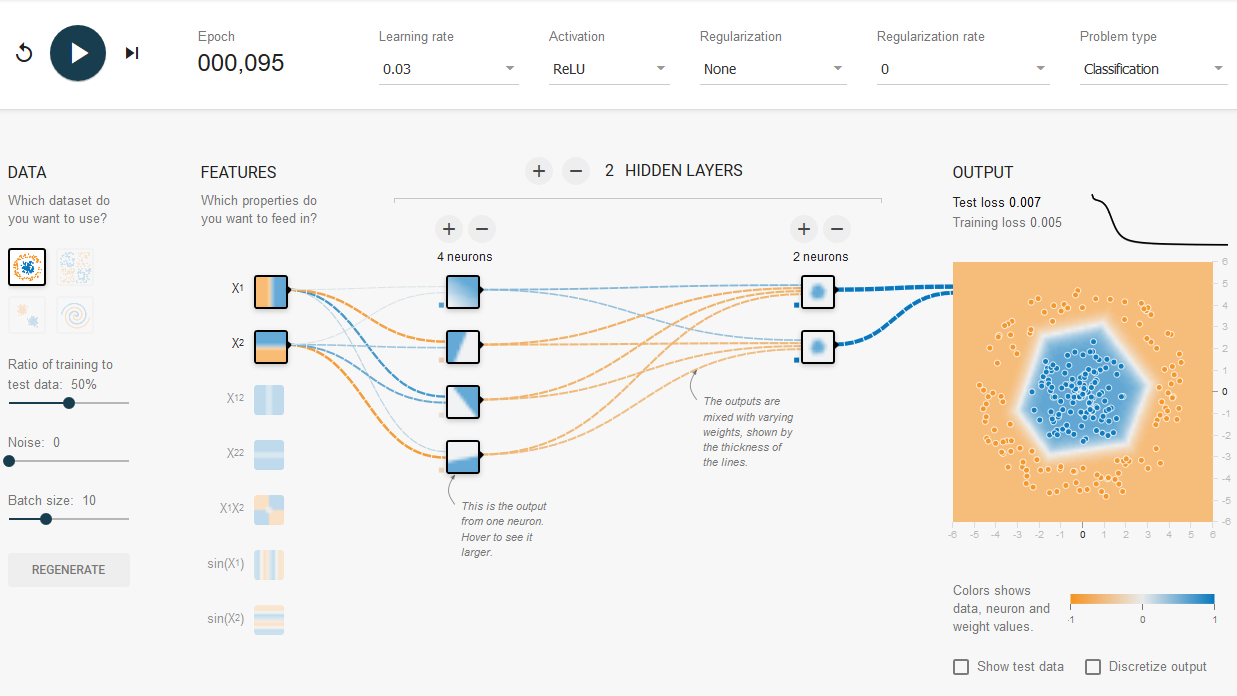


Abbildung 3: Anzahl der Epochen bei Nutzen der ReLU-Funktion -> 95

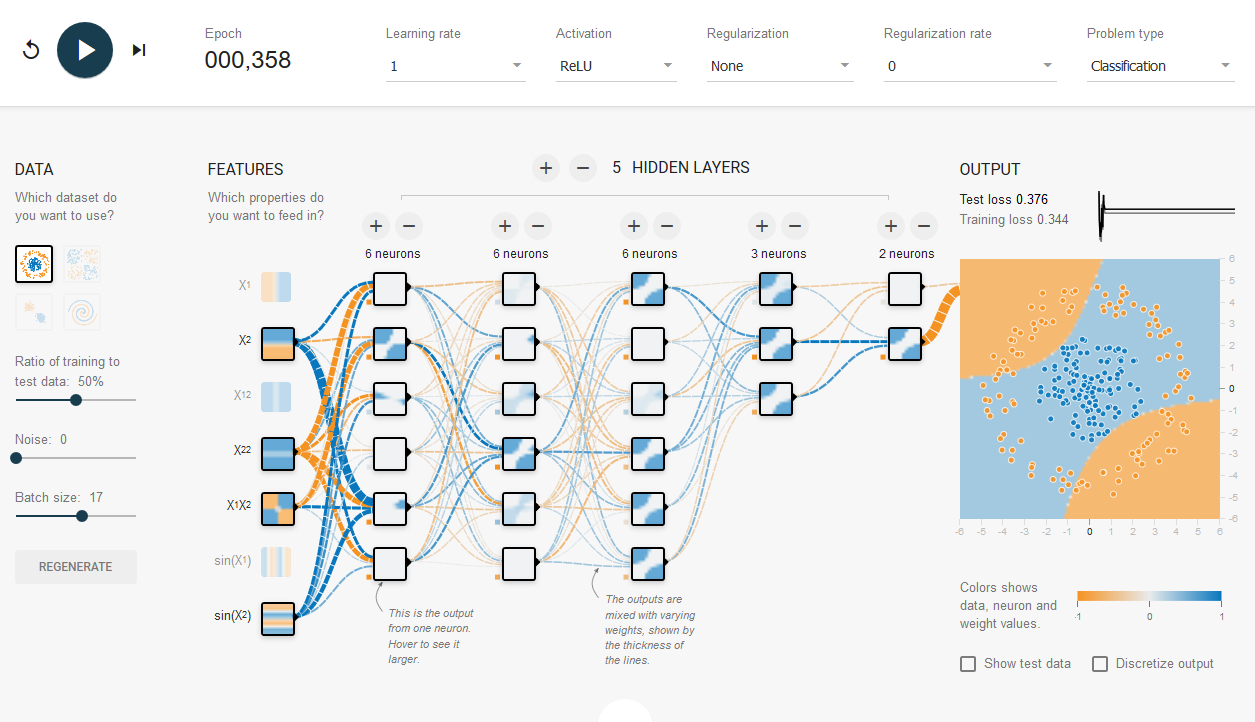


Abbildung 4: Dying ReLU

A1.5

a)

i.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Aktivitäten** | | **Gradient** |
| Mittelwert | Standardabweichung |
| tanh | Konvergiert gegen 0, ist bei ReLU zu Beginn etwas höher | Kovergiert gegen +/- 0.15, Varianz fällt | Erwartungswert und Varianz steigt von erster zur letzten Schicht nur geringfügig |
| ReLU | Konvergiert gegen 0, Varianz fällt |
| ELU | Konvergiert gegen +/-0.11, Varianz fällt |
| SELU | Konvergiert gegen +/-1, Varianz steigt von der ersten bis zur letzten Schicht | Varianz und Erwartungswert fällt |

ii. Der Erwartungswert der ELU und ReLU der Funktionen ist in den ersten Schichten weiter von der 0 entfernt.

iii. die tanh Funktion besitzt in der ersten Schicht eine Standardabweichung von ca. 0.48 und einen Mittelwert von ca 0.

Da der Gradient von tanh im Intervall [-0.48, 0.46] Erwartungswert 0, jedoch eine geringere Standardabweichung als 0.48 hat, sinkt die Varianz der Netzwerkausgaben nur leicht. Der learning slowdown ist also kaum bemerkbar.

iv. am meisten wird mit dem SELU gelernt, denn da sind die Gradienten auch in der letzten Schicht (da in etwa normalverteilt) relativ gesehen am größten.

b) tanh: zu Beginn (layer=1) sind die Aktivierungen symmetrisch um 0 verteilt. Wie in a),iii. erwähnt, werden sowohl die negativen und positiven Aktivierungen von Schicht zu Schicht nach und nach abgeschwächt, bis diese einen Wert von 0 erreichen. Dies erklärt das Maximum der Aktivierungen von 0 in der 20. Schicht.

ReLU: In jeder Schicht werden negative Aktivierungen auf 0 gesetzt. So werden nur noch positive Werte an die nächste Schicht weitergegeben. Es werden also von Schicht zu Schicht Ausgaben „abgeschaltet“ so konvergiert die Ausgabe sehr schnell gegen das lokale Maximum bei 0 Aktivierungen.

ELU: ähnlich wie bei ReLu nur abgeschwächt, da negative Werte nicht direkt auf 0 gesetzt werden sondern nur sehr klein gesetzt werden.

SELU: hier bilden sich zwei lokale Maxima hervor. Das erste bei activations=0. Dies hat den gleichen Grund wie bei (R)ELU, gilt jedoch nur, wenn u>0.