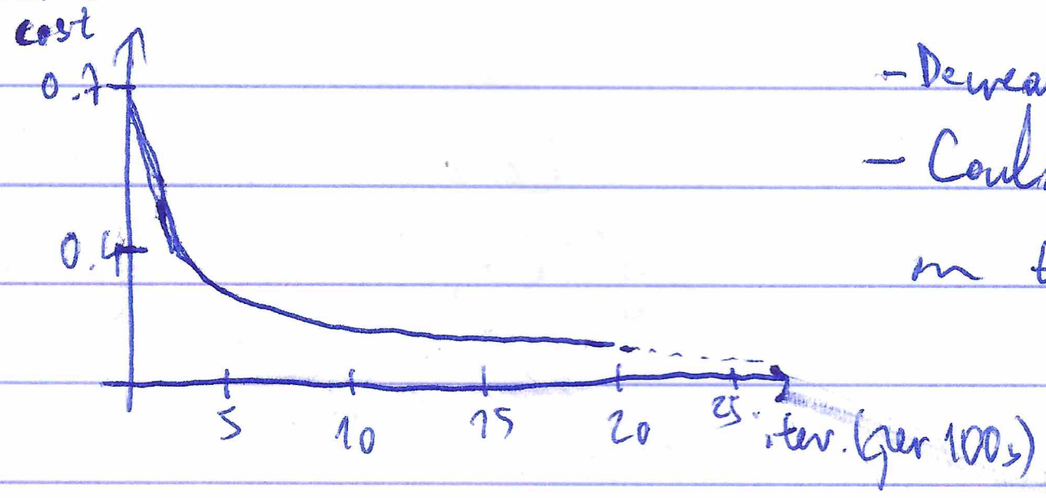
→ - Why np.sum() if vectorised?
→ - Why is transpose .T?
- Optimisation: update parameters using GD
  • Learn w & b by minim. J
  • For a parameter θ, the update rule is $\theta := \theta - \alpha d\theta$

• To use the learned w and b to predict the labels for a dataset X:
  1. Calc. $\hat{Y} = A = \sigma(w^T X + b)$
  2. Convert the entries of a into 0 (if activation ≤ 0.5) or 1 (if activation > 0.5), and store predictions in Y_prediction (vector). Can use if/else in a for-loop (but could also vectorise)
  → - Why not Y_prediction[i] = 0 (or 1)? (instead, [0,i])

- Merge all functions into a model
  • Initialise w, b
  • GD
  • Retrieve (now learned) w & b
  • Predict test/train set examples (Y_prediction_test, Y_prediction_train)

- Comments:
  • Train acc. is close to 100%, ∴ model is working and has high enough capacity to fit the train. data
  • Test error = 68% - not bad
  • Overfitting train. data
  • Cost vs. iterations:



- Decreasing (params. are being learned)
- Could train model even more on train. set (more iters)