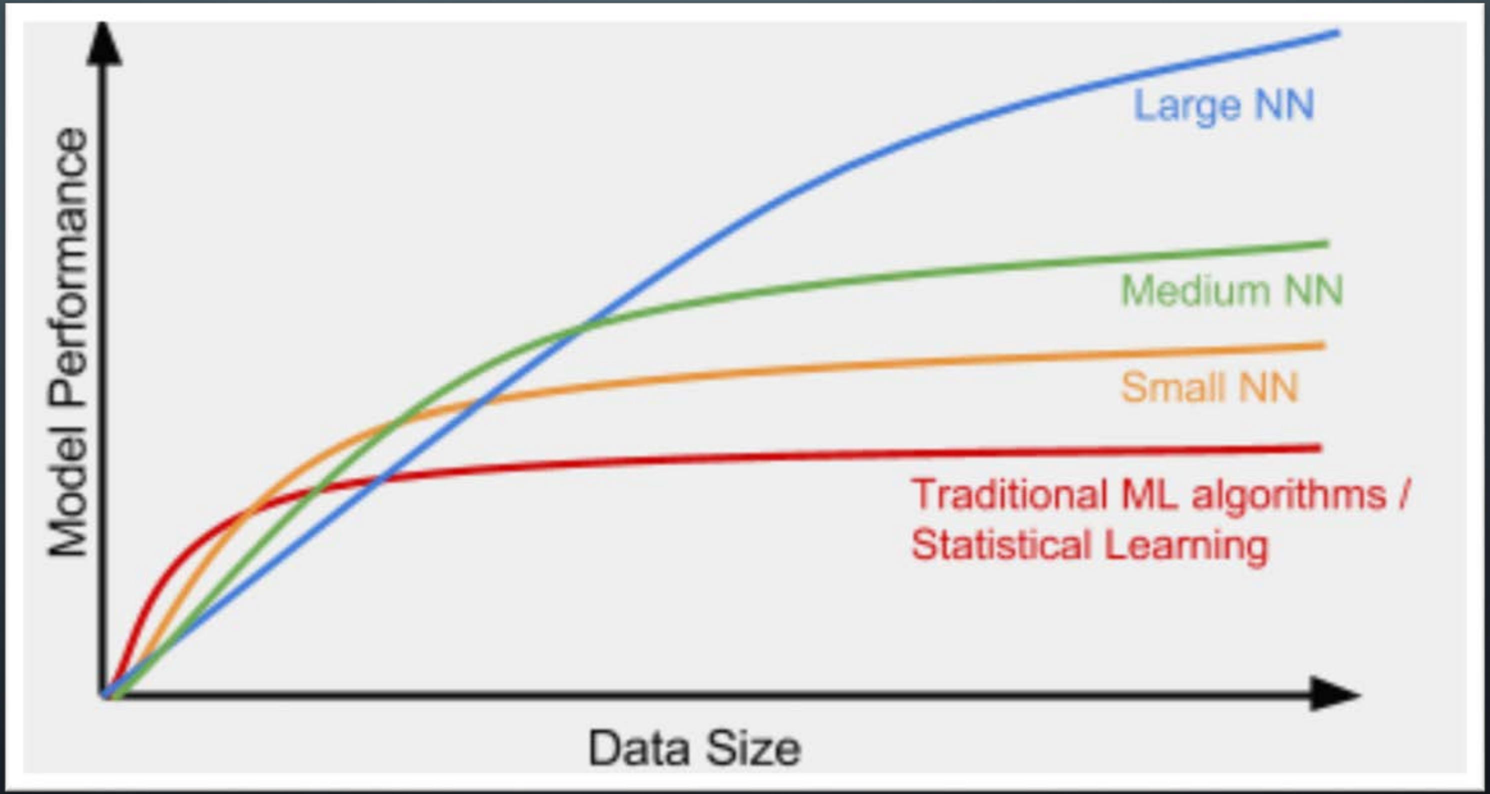
MACHINE LEARNING PROBLEMS

MOHAMMAD GHODDOSI

MACHINE LEARNING MAIN CHALLENGES

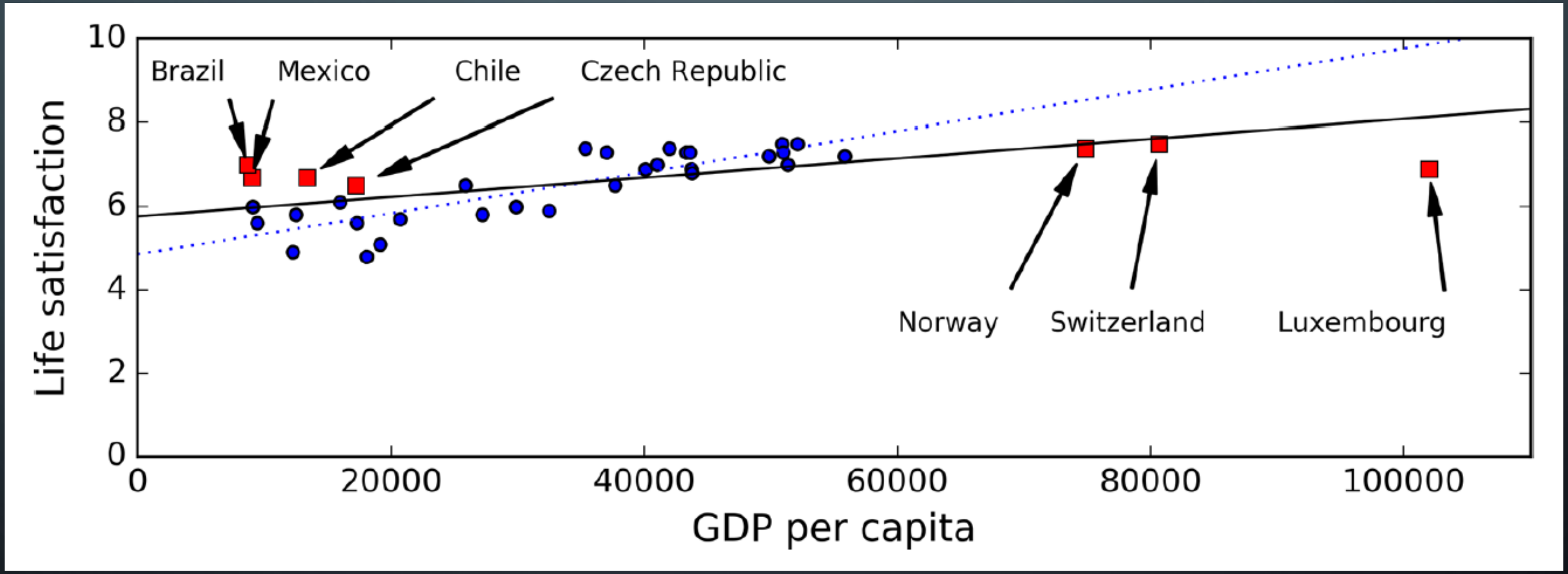
- Insufficient Quantity of Training Data
- Nonrepresentative data
- Poor-quality data
- Irrelevant feature
- Overfitting
- Underfitting

DATA SIZE

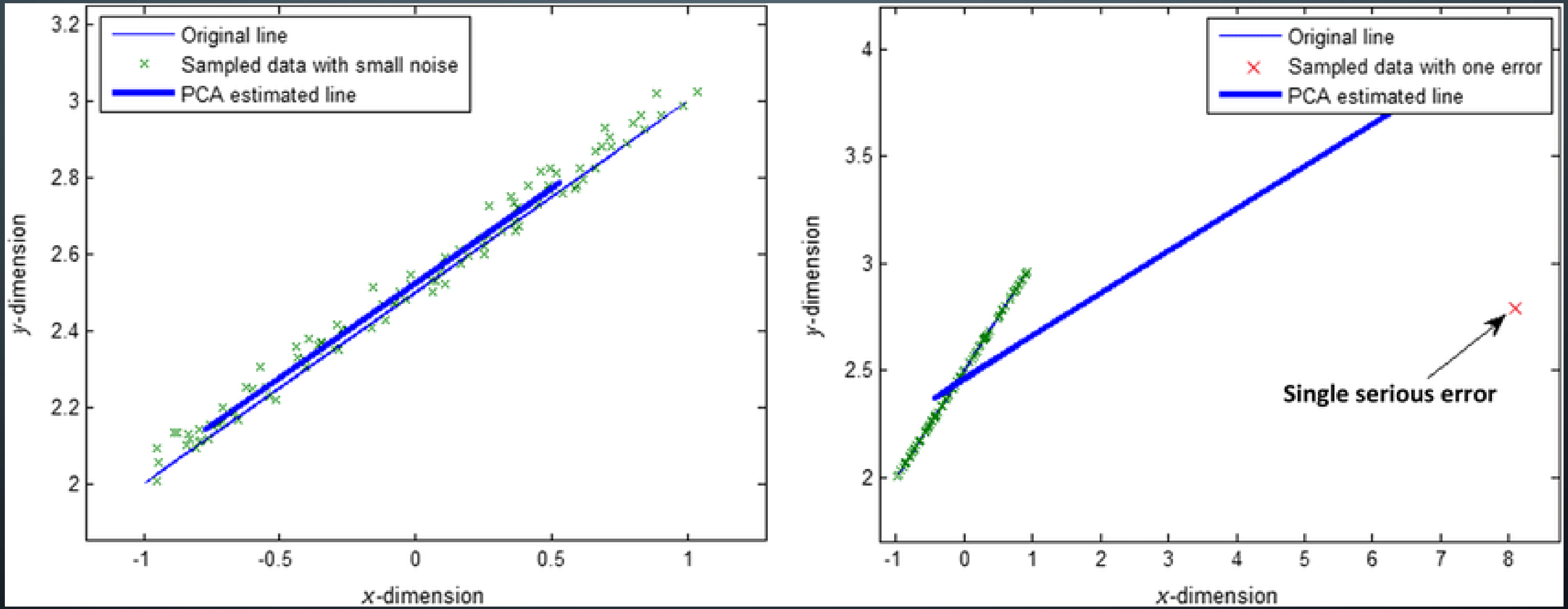


Model Performance
Data Size
Large NN
Medium NN
Small NN
Traditional ML algorithms / Statistical Learning

NONREPRESENTATIVE DATA



POOR-QUALITY DATA



HANDLING DATA QUALITY

- Noise
- Outlier
- Missing value

MISSING VALUES

- Get rid of the row
  - df.dropna()
- Get rid of the attribute
  - df.drop(attribute)
- Set missing values to some value
  - sklearn.preprocessing.Imputer()

SET MISSING VALUES TO SOME VALUE

- Out of range value (label as missing)
- Mean
- Median
- Machine learning models

IRRELEVANT FEATURE

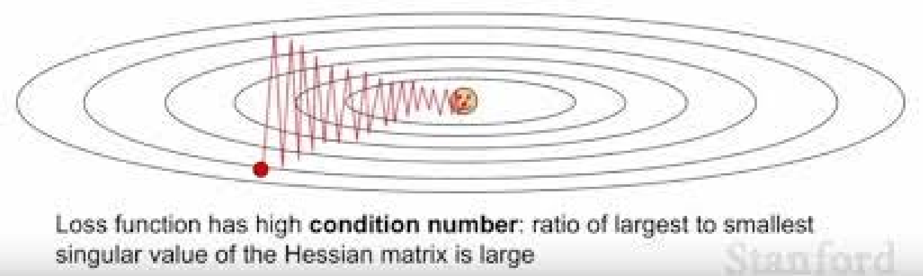
- Garbage in, garbage out
- Feature selection
- Feature extraction
- Gathering new features

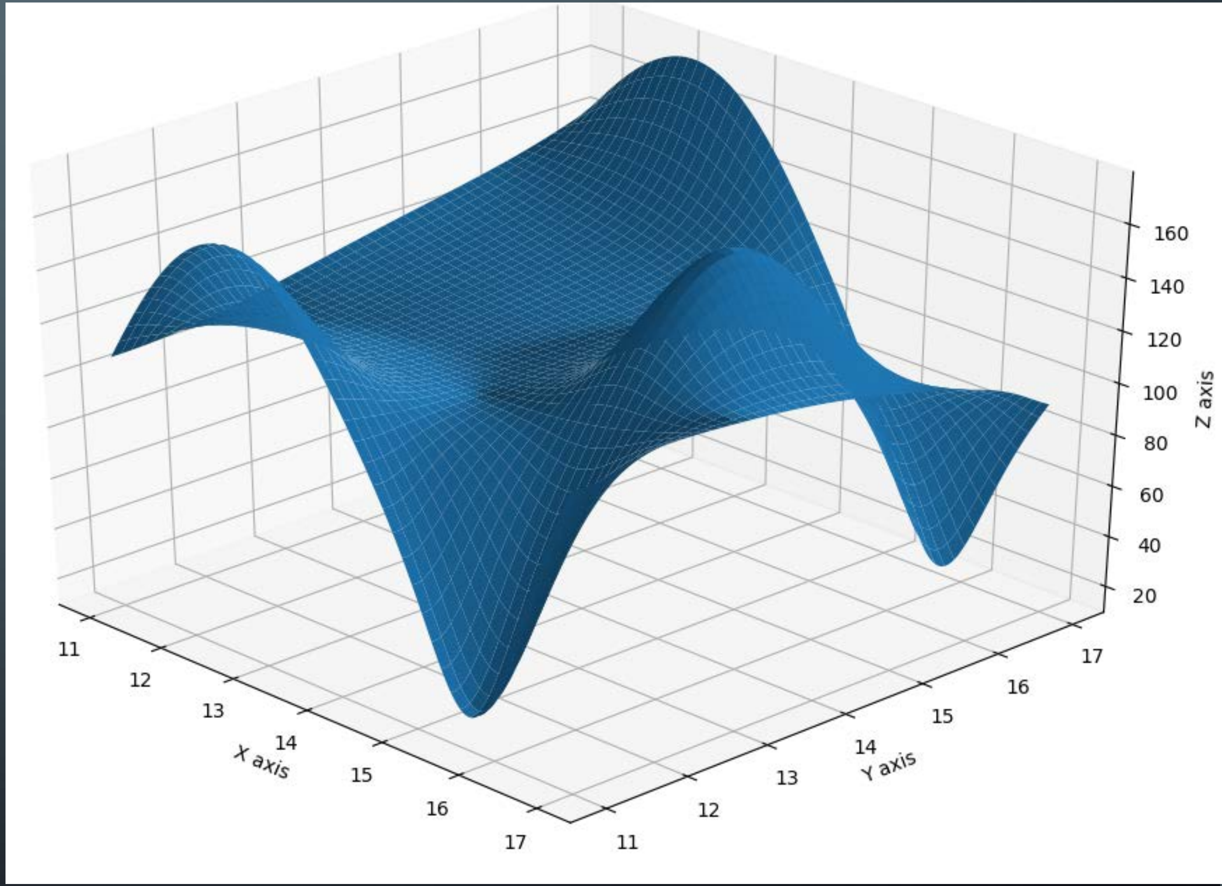
ONLINE / OFFLINE LEARNING

- We don’t have all data
- We need model to update during test time
- Good for dynamic environments

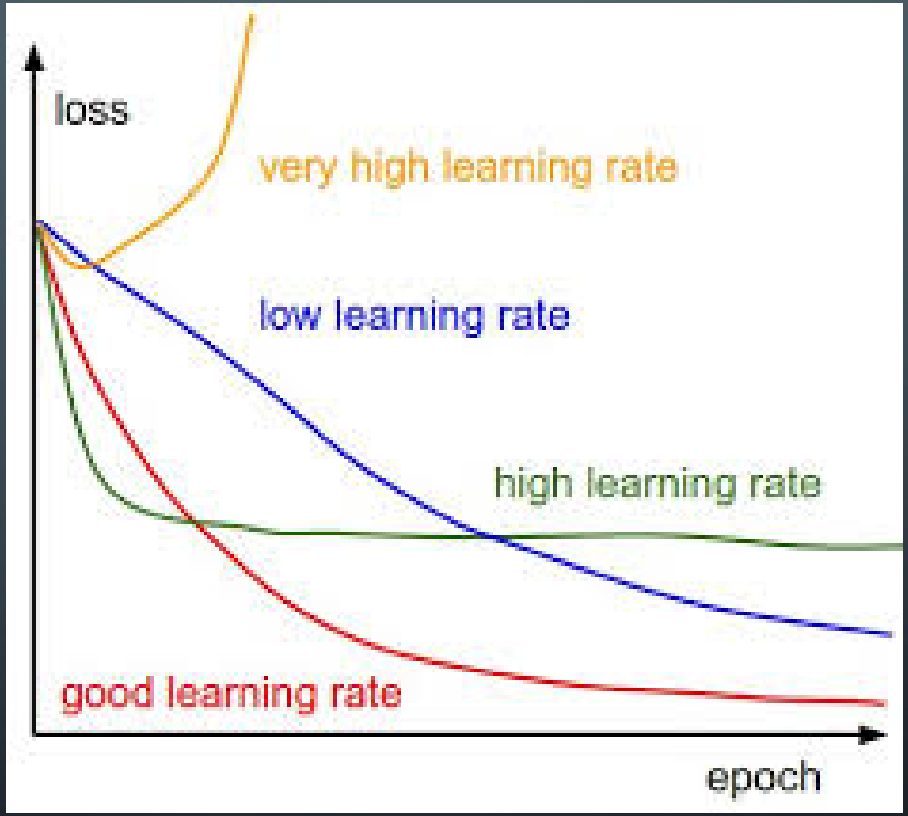
PROBLEMS IN OPTIMIZERS

- Plateau
- Saddle point
- Local minimum
- Zig-zag moves

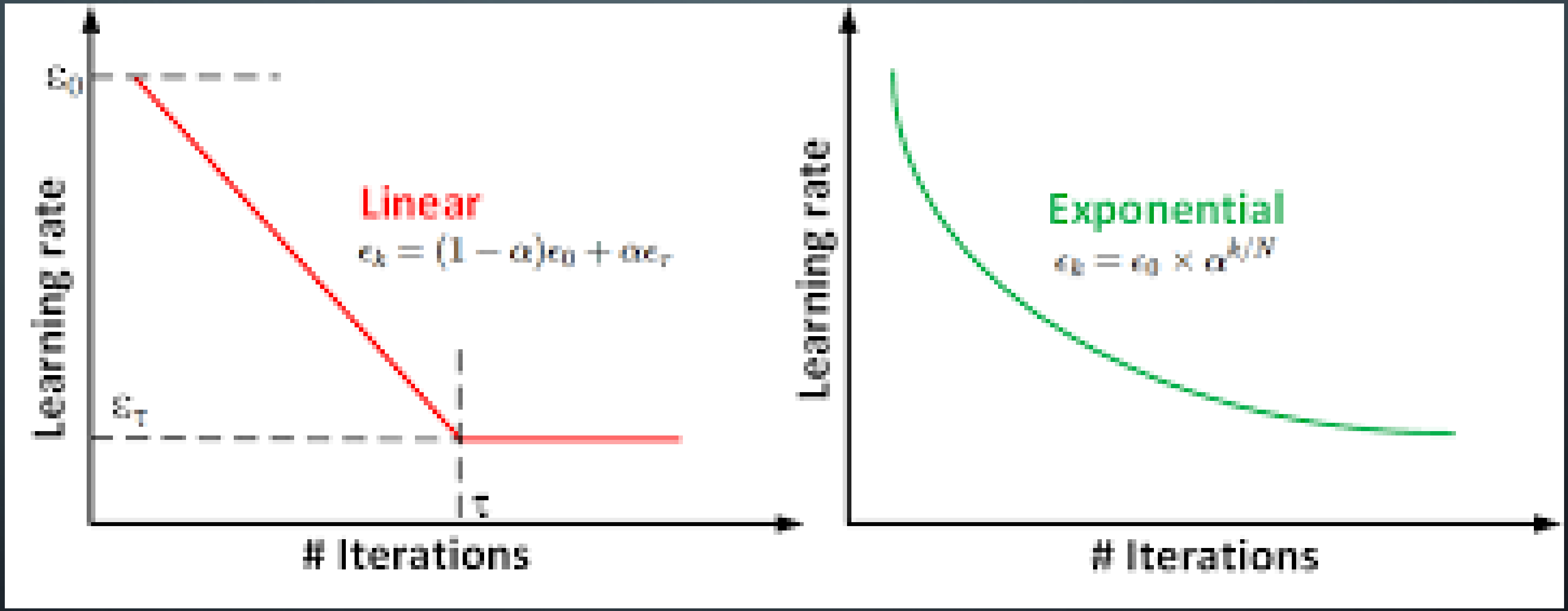




GRADIENT DECENT – LR



GRADIENT DECENT – DECAY LR

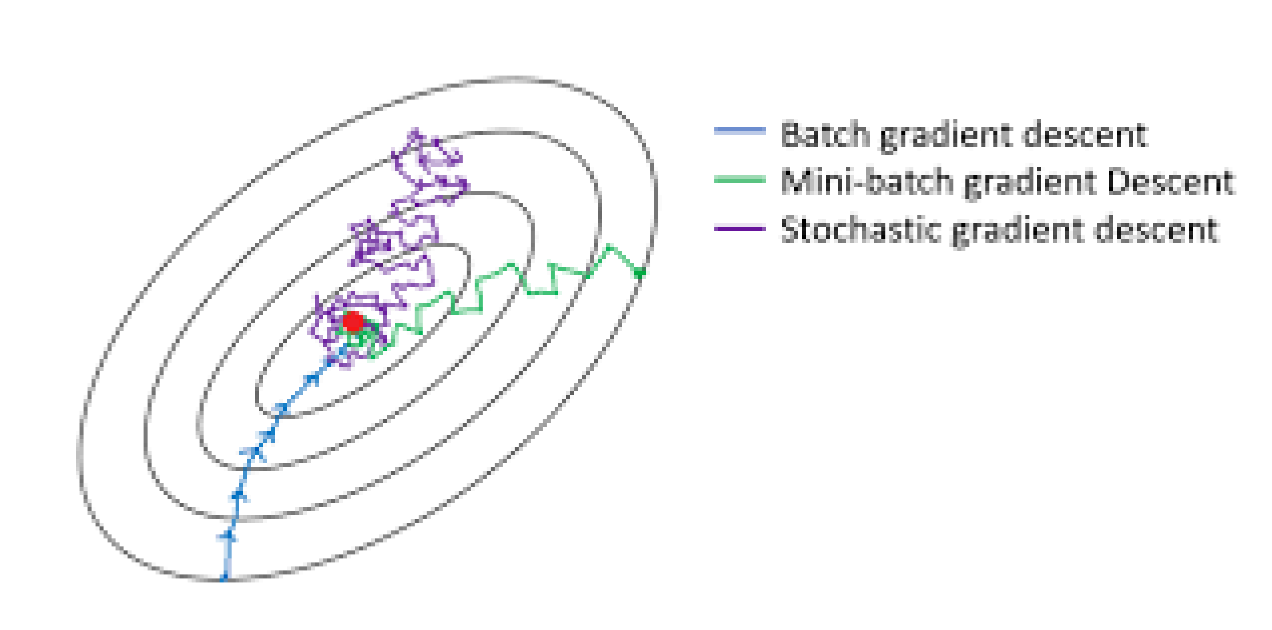


STOCHASTIC GRADIENT DESCENT (SGD)

- Stochastic moves
- avoiding local minimum
- avoiding saddle points
- avoiding plateau
- Computational complexity

MINI-BATCH GRADIENT DESCENT

- Mini-batch
- Not too stochastic
- Fast
- Scalable
- Batch size
- epoch



MOMENTUM

- Like momentum in physics
- Remember update at each step
- Determine next update using
  - Gradient
  - Pervious updates
- avoiding zig-zag moves

MOMENTUM FORMULA

- Normal GD:

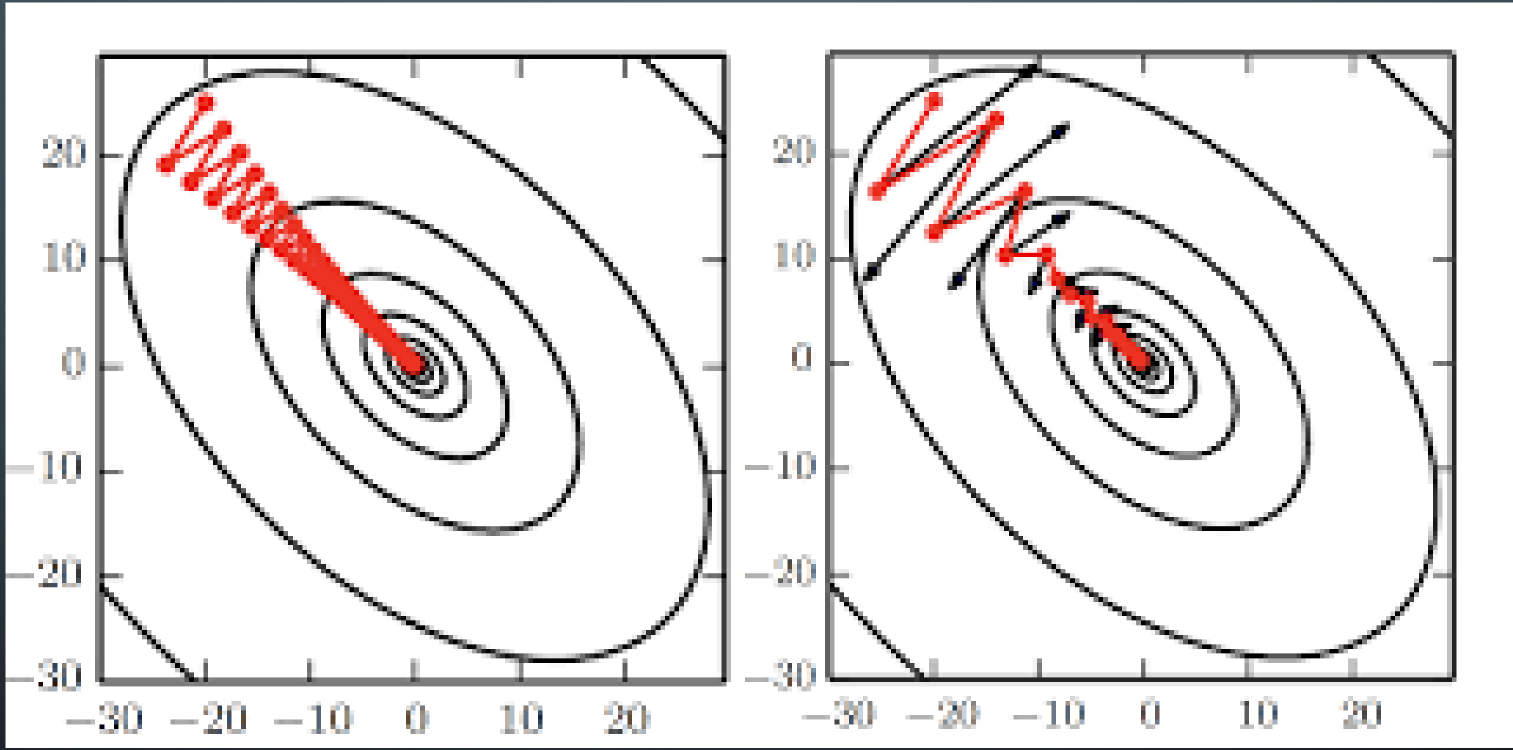
$$\Delta \mathrm{W} = -\eta \nabla J$$

- Momentum GD:

$$v = \alpha v \; - \eta \nabla \mathrm{J}$$

$$\Delta \mathrm{W} = v$$

MOMENTUM EFFECT



ADAPTIVE LEARNING RATE

- Increase or decrees learning rate during training
- different learning rates for different parametters
- AdaDelta
- AdaGrad
- RMSprop
- Adam

ADAGRAD

- scaling learning rates inversely proportional to the square root of the sum of all the historical squared values of the gradient
- For w_i where $\frac{\partial J}{\partial w_i}$ is small, $\Delta\eta_i$ is small
- For w_j where $\frac{\partial J}{\partial w_j}$ is large, $\Delta\eta_j$ is large
- Not good in some nonconvex functions

$$\boxed{\begin{gathered} r = r + (\nabla \mathrm{J} \odot \nabla \mathrm{J}) \\[1em] \Delta \mathrm{W} = -\frac{\eta}{\delta + \sqrt{r}} \nabla \mathrm{J} \end{gathered}}$$

RMSPROP

- Better than AdaGrad in nonconvex functions
- Like AdaGrad but with leakage

$$r = \rho r + (1 - \rho)(\nabla \mathrm{J} \odot \nabla \mathrm{J})$$

$$\Delta \mathrm{W} = -\frac{\eta}{\delta + \sqrt{r}} \nabla \mathrm{J}$$

ADAM

- Using both momentum and RMSprop

$$r = \rho r + (1 - \rho)(\nabla \mathrm{J} \odot \nabla \mathrm{J})$$

$$v = \alpha v - \frac{\eta}{\delta + \sqrt{r}} \nabla \mathrm{J}$$

$$\Delta \mathrm{W} = v$$

VISUALIZATION

- https://emiliendupont.github.io/2018/01/24/optimization-visualization/

EVOLUTIONARY COMPUTING

- Another optimization methods
- Based on generation
- Natural selection
- Survival of the fittest