Higgs analysis with QC

12 / Feb / 2019

Ryu Sawada

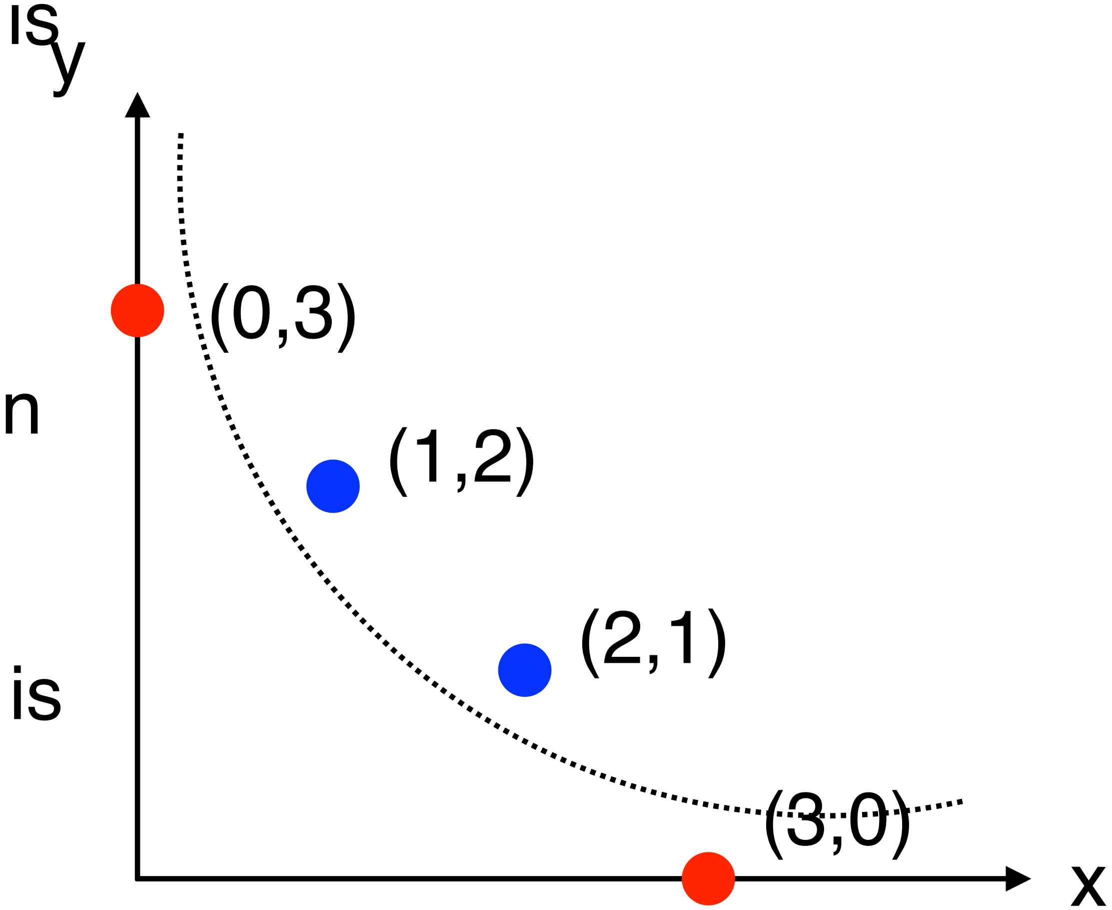
ICEPP, University of Tokyo

Introduction

- As a starting point of using QC for physics analysis, I tried following a presentation at a CERN QC workshop
- The original study by Wen Guan, Winsconsin-Madison
  - ttH, $H \to \gamma\gamma$
  - IBM Qiskit with SVM Quantum Kernel (QSVM)
    - He tried running the program on QC and CPU (simulation).
  - Result
    - Couldn't use enough events on QC due to the limitation of the payload size.
    - The execution in Simulator is limited by the memory size.
      - 34 GB for 31 qubits
      - 47 GB for 8 qubits for 200 events with full entanglement

Support Vector Machine

- Support vector machine (SVM) is a supervised machine learning algorithm to classify data (set of variables) into two or more classes.
- Internally, it makes higher dimensional "feature" space to separate non-linearly.
  - For example, if we define a new axis as $z = x \times y$, we can separate the data shown below by a plane $z = 1$.
- The complexity of classical SVM algorithms is approximately proportional to N^3.
- Quantum version can be proportional to log(N) by,
  - performing inner products of the vectors in parallel
  - converting the SVM training to an approximate least-square problem which is subsequently solved it by the quantum matrix inversion algorithm



arXiv:1307.0471 [quant-ph] , arXiv:1410.1054 [quant-ph]

R.Sawada

Data used this time

- Kaggle Higgs ML Challenge data
  - Signal : H→$\tau\tau$ (had τ + lep τ)
  - BG : ttbar, Z, W which contains τ in the final states.
  - Training data
    - ~30 variables
    - Signal : ~90k events, BG : ~160k events

- c.f.: results in the ML challenge
  - The best one uses Gradient Boost Classifier with the success rate of 84%
  - The best SVM result was 76%.

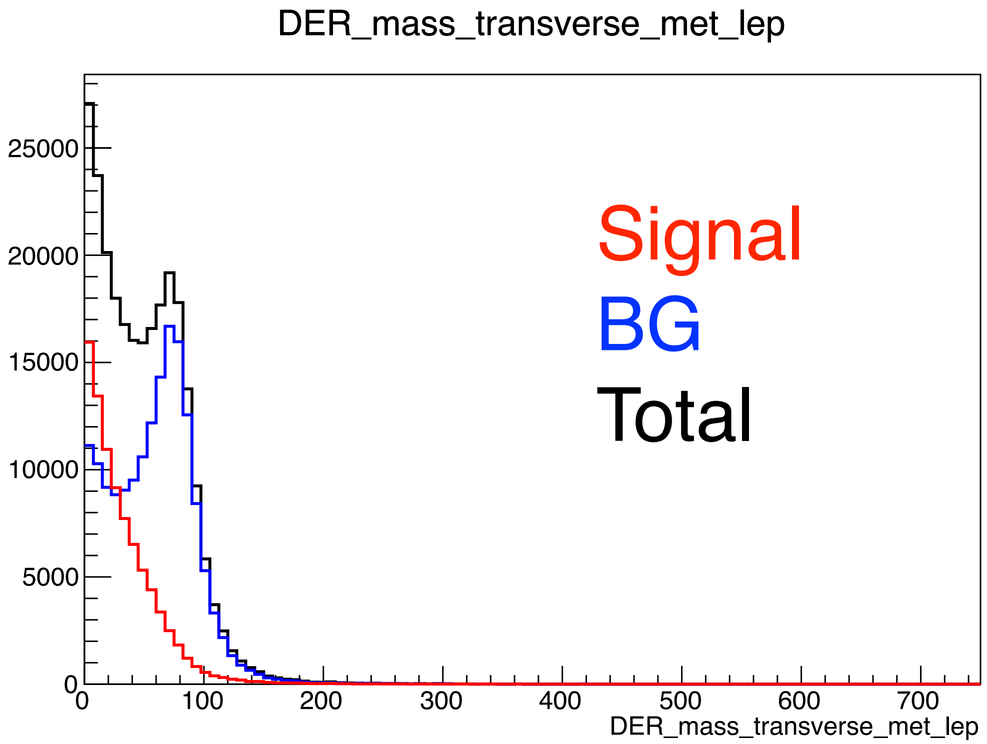
The algorithm

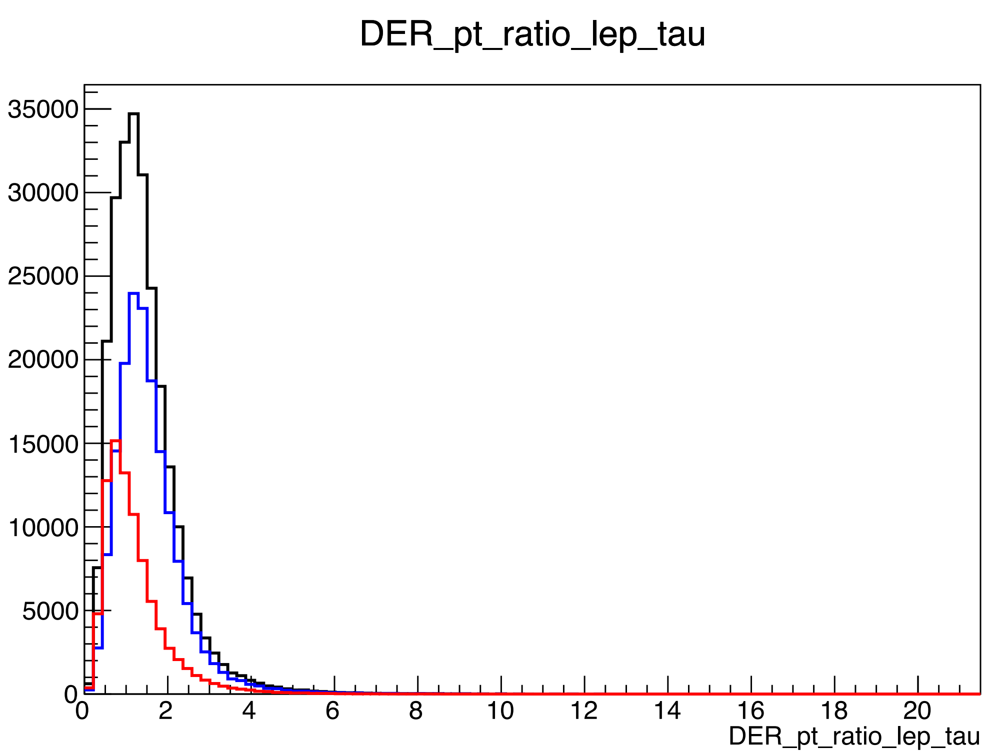
- Based on the QSVM code in an example code in Qiskit.
- In QSVM the number of classical feature (= input variables) is equal to the number of qubits.
- So far I used only two inputs (should be able to increase).

Results

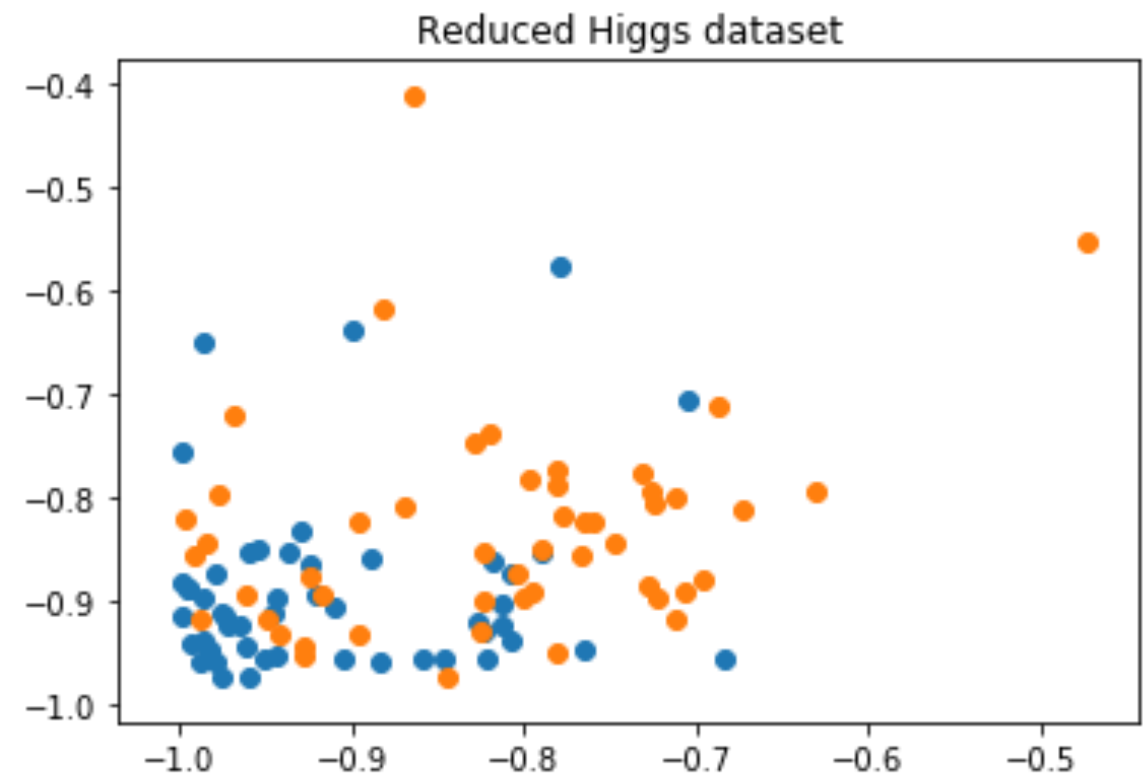
I selected two most-separating variables.

All sample





QSVM training with 50 events



(The values are scaled)

The success rate of 20 events: 62.5 %

R.Sawada

Test on IBM Q

- On real QC (IBM Q 16 Melbourne),
  - With 50 training sample
    - Couldn’t run with a “GENERIC ERROR: 400”, maybe due to too large payload ?
    - By reducing the number of training sample to 1, the error has gone, but the training didn’t end in ~5 hours…
- On HPC (IBM Q QASM Simulator)
  - It runs successfully in 20 minutes.
- On a local computer with simulation (Intel silver 10 core 2.2GHz)
  - It runs successfully in 10 minutes.

Summary

- QSVM can be a good algorithm to use on existing QC because it does not require too many qubits.
  - The number of qubits is equal to the number of input variables.
- We can start using QSVM algorithm for physics analyses using an example code in Qiskit.
- However, it does not seem to be possible run it on “public” IBM Q machines due to limited payload size (?).
- In order to do practical analysis, we probably need much higher number of training samples.
  - How about other QC
    - Other IBM Q machines (like Tokyo or Austin) which are reserved for members of IBM Q network
    - Google ?
    - Rigetti ?

Backup

Variable list

EventId
DER_mass_MMC
DER_mass_transverse_met_lep
DER_mass_vis
DER_pt_h
DER_deltaeta_jet_jet
DER_mass_jet_jet
DER_prodeta_jet_jet
DER_deltar_tau_lep
DER_pt_tot
DER_sum_pt
DER_pt_ratio_lep_tau
DER_met_phi_centrality
DER_lep_eta_centrality
PRI_tau_pt
PRI_tau_eta
PRI_tau_phi

PRI_lep_pt
PRI_lep_eta
PRI_lep_phi
PRI_met
PRI_met_phi
PRI_met_sumet
PRI_jet_num
PRI_jet_leading_pt
PRI_jet_leading_eta
PRI_jet_leading_phi
PRI_jet_subleading_pt
PRI_jet_subleading_eta
PRI_jet_subleading_phi
PRI_jet_all_pt
Weight
Label