# Financial Wellbeing

## Using quantum machine learning to identify people who are in need of financial support.

**Qiskit Hackathon: July 2022**

Team*: Anthony Hsu (BMM Lab), Ming Chen (CS) , Nitin Yadav (BMM Lab)
Coach: Desiree (IBM) and Charles (Physics)

*Thanks to Udaya for support.

# Financial Wellbeing

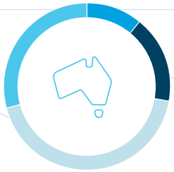
**Usefulness and Complexity**

## FINANCIAL WELLBEING IN AUSTRALIA AT A GLANCE

### THE FOUR COMPONENTS OF FINANCIAL WELLBEING

- Meeting everyday commitments
- Feeling comfortable
- Financial resilience
- Feeling secure for the future

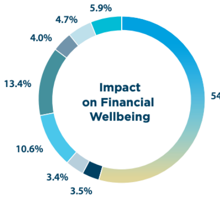
### FINANCIAL WELLBEING CATEGORIES IN AUSTRALIA




### FINANCIAL WELLBEING SCORE (OUT OF 100)

66 Male vs 62 Female

### WHAT CAN INFLUENCE FINANCIAL WELLBEING?




- 54.5% Socio-economic conditions
- 3.5% Knowledge & experience
- 3.4% Financial confidence & control
- 10.6% Saving & spending behaviours
- 13.4% Behaviour traits
- 4.0% Money management behaviour
- 4.7% Investment behaviour
- 5.9% Saving & spending attitudes

**Socio-economic conditions have the largest influence on financial wellbeing**

POSITIVE EFFECT

- 7.5% Earning potential
- 5.9% Life journey
- 5.5% Access to social support
- 5.2% Financial stability
- 4.6% Male gender

NEGATIVE EFFECT

- 13.7% Health concerns
- 8.5% Unemployment
- 3.6% Financial support to a dependent

### SAVING, SPENDING AND INVESTMENT BEHAVIOURS IMPORTANT FOR FINANCIAL WELLBEING

- Active saving
- Not borrowing for everyday expenses
- Spending restraint
- Investment

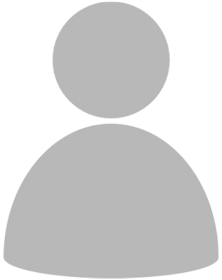
Source: ANZ Financial Wellbeing Survey 2021

# Classification task on real world data



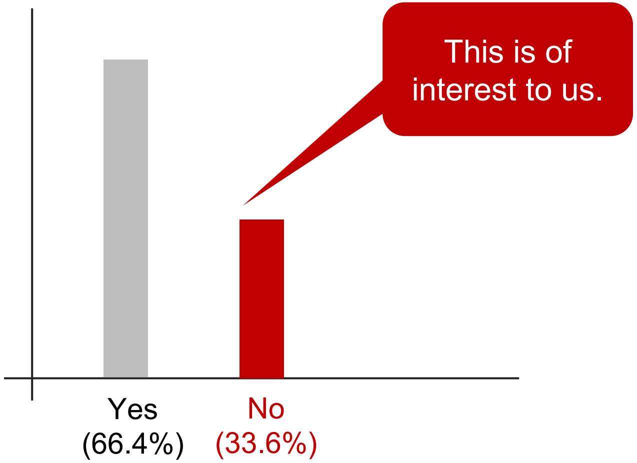
1. Demographics (62+, minority, general population, etc).
2. Living status (alone, with partner, shared, etc).
3. Financial skill score.
4. Ability to make complex decisions.
5. Understanding of compound interest.
6. Understanding of relation between inflation and savings interest rate.
7. Understanding of difference between stock and mutual fund volatility.

Social Context



Financial Context

Financial Wellbeing?




Data available from consumerfinance.gov

# Results: Classical vs Quantum



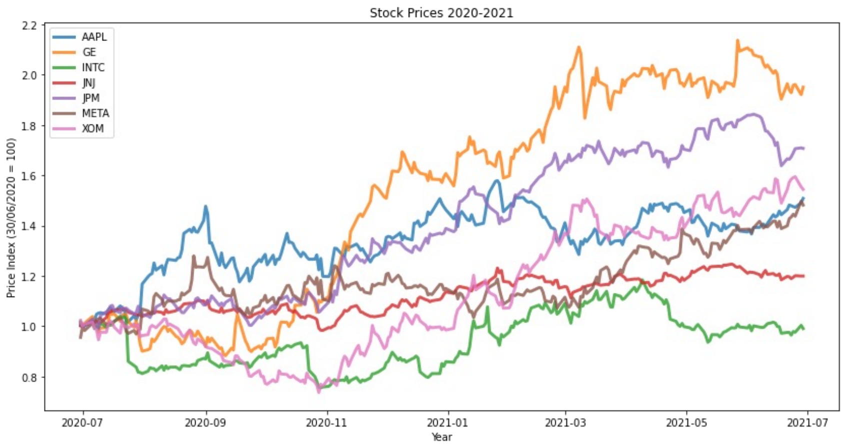
| | AUC | Precision | Recall | F1-Score | Time(s) |
|---|---|---|---|---|---|
| AdaBoost | 0.597 | 0: 0.45<br>1: 0.73 | 0: 0.56<br>1: 0.64 | 0: 0.50<br>1: 0.68 | 0.006 |
| Neural Networks<br>(3 hidden layers) | 0.67 | 0: 0.79<br>1: 0.75 | 0: 0.41<br>1: 0.94 | 0: 0.54<br>1: 0.83 | 0.06 |
| Logit | 0.61 | 0: 0.73<br>1: 0.71 | 0: 0.30<br>1: 0.94 | 0: 0.42<br>1: 0.81 | 0.016 |
| SVM | 0.64 | 0: 0.89<br>1: 0.72 | 0: 0.30<br>1: 0.98 | 0: 0.44<br>1: 0.83 | 0.007 |
| Quantum VQC<br>(ZZFeature, RealAmplitudes) | 0.55 | 0: 0.41<br>1: 0.69 | 0: 0.48<br>1: 0.62 | 0: 0.44<br>1: 0.65 | 912.5 |
| Quantum Kernel<br>(ZZFeature, linear entl.) | 0.55 | 0: 0.50<br>1: 0.68 | 0: 0.22<br>1: 0.88 | 0: 0.31<br>1: 0.77 | 954.4 |
| Quantum Kernel<br>(ZFeature) | 0.60 | 0: 0.56<br>1: 0.71 | 0: 0.37<br>1: 0.84 | 0: 0.44<br>1: 0.77 | 618.2 |
| Quantum Kernel<br>(PauliFeature) | 0.48 | 0: 0.25<br>1: 0.64 | 0: 0.07<br>1: 0.88 | 0: 0.11<br>1: 0.74 | 2144.75 |

Ran on 6% of the dataset. Core i9-10900F@4.8Ghz, 64GB DDR4 3466 MHz, Windows 11 Pro under Jupyter Notebook environment.  QML was run on a simulator.

# Portfolio optimisation on real world data

## Just for fun…

**Bonus?**




| **Metrics** | Expected Return | Expected Risk | Sharpe Ratio |
|---|---|---|---|
| Equally weighted | 55.02% | 20.94% | 2.46 |
| Classical (scipy, slsqp) | 71.44% | 23.02% | 2.95 |
| VQE | 63.3% | 21.28% | 2.81 |
| QAOA | 76.34% | 25.25% | 2.89 |

| **Weights** | AAPL | META | XOM | JNJ | JPM | INTC | GE | TSLA |
|---|---|---|---|---|---|---|---|---|
| Equally weighted | 0.12 | 0.12 | 0.12 | 0.12 | 0.12 | 0.12 | 0.12 | 0.12 |
| Classical | 0.00 | 0.25 | 0.00 | 0.05 | 0.30 | 0.24 | 0.16 | 0.00 |
| VQE | 0.10 | 0.30 | 0.00 | 0.10 | 0.10 | 0.20 | 0.10 | 0.10 |
| QAOA | 0.10 | 0.30 | 0.00 | 0.00 | 0.20 | 0.20 | 0.20 | 0.00 |

*Note: Only one year of past daily data taken and hence Sharpe ratio is high.*

# What did we learn?



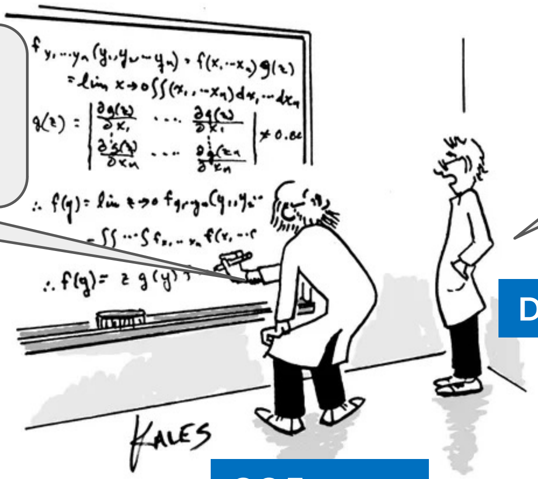
- Hackathons are fun and an *efficient* way to get into something new.
- Double (triple) check the qiskit doc. version and the installed version.
- Inputs to the VQC and Kernel methods may be different.
  - The target variable needed one-hot encoding in VQC but not in kernel methods.
- Different runs of a QML algorithm may yield diverse performance
  - Hyper parameter tuning is based on experience with these methods.
- QML methods are (currently) magnitudes slower than classical ML methods.
  - Implementing parallel processing in simulators can be tricky.
  - One needs patience (a lot of it…)
- QML methods (currently) have a lower performance than classical ML methods.
- One has to be aware of issues arising from using small datasets (in the presence of noise).

# Healthy cross-disciplinary collaboration



With advances in Quantum Computing we will be able to provider better solution methods, so that you can make better decisions.

OK. But I don’t understand how to use these methods (what is entanglement?)!




Decision Scientist

QC Expert

Thank you to UoM Physics and IBM Quantum group for organizing this event.

# *Qiskit-AI based Prediction for Financial Wellbeing*

Speaker: Ming Chen
mingc4@student.unimelb.edu.au

Team*: Anthony Hsu (BMM Lab), Ming Chen (CS) , Nitin Yadav (BMM Lab) Coach: Desiree (IBM) and Charles (Physics)

THE UNIVERSITY OF MELBOURNE

# Problem Introduction

**Financial wellbeing** involves measuring people's financial comfort in the context of their day to day expenses, future savings (e.g., retirement) and their resilience to financial shocks.

The plan is to **train quantum machine learning algorithm**(s) over **real world financial wellbeing data** with the aim to better model financial wellbeing measure given the parameters.

```
df_raw
```

✓ 0.6s

| | fwb | sample | fs_score | fs_assess | fk_interest | fk_inflation | fk_vola | living |
|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 44 | 4 | 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 43 | 3 | 1 | 1 | 1 | 2 |
| 2 | 0 | 1 | 42 | 3 | 1 | 1 | 1 | 2 |
| 3 | 0 | 1 | 42 | 3 | 1 | 0 | 1 | -1 |
| 4 | 0 | 1 | 42 | 3 | 0 | 0 | 1 | 3 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 6384 | 1 | 3 | 47 | 3 | 1 | 1 | 1 | 2 |
| 6385 | 1 | 3 | 59 | 4 | 1 | 0 | 0 | 2 |
| 6386 | 1 | 1 | 51 | 3 | 1 | 0 | 1 | 2 |
| 6387 | 0 | 1 | 54 | 4 | 1 | 0 | 1 | 2 |
| 6388 | 0 | 3 | 42 | 3 | 1 | 0 | 1 | 2 |

6389 rows × 8 columns

THE UNIVERSITY OF MELBOURNE

# Problem Solving

Model used-

`from qiskit_machine_learning.algorithms import` **PegasosQSVC**

```python
from qiskit_machine_learning.algorithms import PegasosQSVC

# should test with different (tau, C), to find the optimal scores.
# Doing grid search

print("PegasosQSVC classification grid search param set and mean accuracy: ")
count = 1

for C in qSVC_param_grid['C_grid']: # regulaztion param
    for tau in qSVC_param_grid['tau_grid']: # iterations
        pegasos_qsvc = PegasosQSVC(quantum_kernel=qkernel, C=C, num_steps=tau)
        # training
        pegasos_qsvc.fit(train_features, train_labels)

        # testing
        # Output the mean accuracy on the given test data and labels.
        pegasos_score = pegasos_qsvc.score(test_features, test_labels)

        print("ParmaSet@" + str(count))
        print(f"C : {C},  tau : {tau}, test_accuracy : {pegasos_score}")
        count+=1

        #print(f" test score: {pegasos_score}")
```

Pegasos: Primal Estimated sub-GrAdient SOlver for SVM

INPUT: $S, \lambda, T$
INITIALIZE: Set $\mathbf{w}_1 = 0$
FOR $t = 1, 2, \ldots, T$
  Choose $i_t \in \{1, \ldots, |S|\}$ uniformly at random.
  Set $\eta_t = \frac{1}{\lambda t}$
  If $y_{i_t} \langle \mathbf{w}_t, \mathbf{x}_{i_t} \rangle < 1$, then:
    Set $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t \lambda)\mathbf{w}_t + \eta_t y_{i_t} \mathbf{x}_{i_t}$
  Else (if $y_{i_t} \langle \mathbf{w}_t, \mathbf{x}_{i_t} \rangle \geq 1$):
    Set $\mathbf{w}_{t+1} \leftarrow (1 - \eta_t \lambda)\mathbf{w}_t$
  [ Optional: $\mathbf{w}_{t+1} \leftarrow \min\left\{1, \frac{1/\sqrt{\lambda}}{\|\mathbf{w}_{t+1}\|}\right\} \mathbf{w}_{t+1}$ ]
OUTPUT: $\mathbf{w}_{T+1}$

# Experimental Results

## *Optim Methods ----  Circuit Encoding + Grid Search :*

being.ipynb | QSVM_example.ipynb | QSVM_FWB.ipynb | QSVM_FW

```
C : 2000,  tau : 20, test_accuracy : 0.6580594679186228
ParmaSet@26
C : 2000,  tau : 50, test_accuracy : 0.6807511737089202
ParmaSet@27
C : 2000,  tau : 100, test_accuracy : 0.6369327073552425
ParmaSet@28
C : 2000,  tau : 200, test_accuracy : 0.6917057902973396
ParmaSet@29
C : 2000,  tau : 400, test_accuracy : 0.701095461658842
ParmaSet@30
C : 2000,  tau : 1000, test_accuracy : 0.7034428794992176
ParmaSet@31
C : 5000,  tau : 20, test_accuracy : 0.4577464788732394
ParmaSet@32
C : 5000,  tau : 50, test_accuracy : 0.6697965571205008
ParmaSet@33
C : 5000,  tau : 100, test_accuracy : 0.6697965571205008
ParmaSet@34
C : 5000,  tau : 200, test_accuracy : 0.6697965571205008
ParmaSet@35
C : 5000,  tau : 400, test_accuracy : 0.7339593114241002
ParmaSet@36
C : 5000,  tau : 1000, test_accuracy : 0.6744913928012519
ParmaSet@37
C : 20000,  tau : 20, test_accuracy : 0.6025039123630673
ParmaSet@38
C : 20000,  tau : 50, test_accuracy : 0.6580594679186228
ParmaSet@39
C : 20000,  tau : 100, test_accuracy : 0.6964006259780907
ParmaSet@40
C : 20000,  tau : 200, test_accuracy : 0.7089201877934272
ParmaSet@41
C : 20000,  tau : 400, test_accuracy : 0.5477308294209703
ParmaSet@42
```

### Data encoding circuits

These `BlueprintCircuit` encode classical data in quantum states and are used as feature maps for cl

| | |
|---|---|
| `PauliFeatureMap`([feature_dimension, reps, ...]) | The Pauli Expansion circuit. |
| `ZFeatureMap`(feature_dimension[, reps, ...]) | The first order Pauli Z-evolution circuit. |
| `ZZFeatureMap`(feature_dimension[, reps, ...]) | Second-order Pauli-Z evolution circuit. |
| `StatePreparation`(params[, num_qubits, ...]) | Complex amplitude state preparation. |

*Best Result till now----*
*featureMap: ZFeatureMap()*
*C : 5000,*
*tau : 400,*
*test_accuracy : 0.7339593114241002*