Enhancing Predictive Modeling in the Premier League with Parameterized Quantum Circuits (PQCs)

Abdul Haleem Abdul Salam

Mohanad Alkhalaf

Yu Jiyun Chang

Sai Nityamani Sahith Matsa

Yang Ren

Contents

Introduction to **Predictive Modeling**

Definition:

Predictive modeling uses statistical and machine learning techniques to forecast outcomes based on historical data

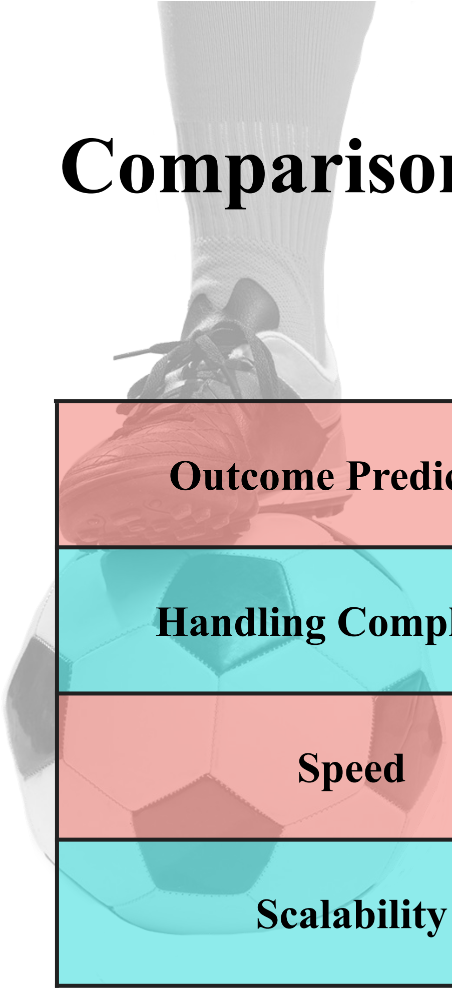
Applications in Football:

- Match Outcome Predictions
- Player Performance Forecasting
- Team Strategy Optimization

Importance in Business:

- Helps clubs make strategic decisions
- Engages fans through fantasy football and betting platforms

Comparison Table



	Classical	Quantum
Outcome Prediction	Deterministic	Probabilistic
Handling Complexity	Limited	Superior
Speed	Slower for large datasets	Potentially faster
Scalability	Limited by classical hardware	Promising with quantum tech

Premier League

Team Selection Liverpool

YOU'LL NEVER WALK ALONE
LIVERPOOL
FOOTBALL CLUB
EST·1892

Quiz 4 Dataset

- Data filtered by Liverpool home and away games
- Games filtered by win or lose
- Games where the result has been draw is removed
- Season 2013-2014 has been considered for this dataset

Final Project Dataset

- More features inclusion such as attack strength and defensive strength for quantum machine learning translation

Overview of Classical vs Quantum Methods

Classical Methods

- Use traditional algorithms like Logistic Regression, Random Forests, Neural Networks
- Sequential data processing

```
# Define the feature set and labels
X = liverpool_matches[['liverpool_home_away', 'opponent_team_mapped']]
y = liverpool_matches['liverpool_result']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

# Initialize and train the Random Forest classifier
rf_clf = RandomForestClassifier()
rf_clf.fit(X_train, y_train)

# Predict the results on the test set
y_pred_rf = rf_clf.predict(X_test)
```

Quantum Methods

- Leverage quantum properties like superposition and entanglement
- Enable parallel processing and handle more complex datasets

```
def classify_match(home_away, opponent_team_mapped, label):
    # Map input features to theta values
    # θ1 is based on ay status
    theta1_value = home_away * np.pi / 2
    # θ2 is based on opponent team mapped to [2,20]
    theta2_value = opponent_team_mapped * np.pi / 5
    # θ3 is the sum of home/away and opponenthome/aw
    theta3_value = (home_away + opponent_team_mapped) * np.pi / 4

    # Create a new quantum circuit with the mapped values
    bound_qc = QuantumCircuit(1)
    bound_qc.rx(theta1_value, 0)  # θ1
    bound_qc.ry(theta2_value, 0)  # θ2
    bound_qc.rz(theta3_value, 0)  # θ3
    bound_qc.measure_all()
```

Quiz 4 - Quantum Model Example

Key Steps:

- Use one qubit with parameterized gates (rx, ry, rz) based on match features (PQC)
- Map rotation angles to the features of the match
  - θ1: Home/Away status
  - θ2: Opponent team
  - θ3: Combined features
- Execute the circuit using qasm_simulator
- Classify the match result
- Evaluate model's performance

```python
def classify_match(home_away, opponent_team_mapped, label):
    # Map input features to theta values
    # θ1 is based on ay status
    theta1_value = home_away * np.pi / 2
    # θ2 is based on opponent team mapped to [2,20]
    theta2_value = opponent_team_mapped * np.pi / 5
    # θ3 is the sum of home/away and opponenthome/aw
    theta3_value = (home_away + opponent_team_mapped) * np.pi / 4

    # Create a new quantum circuit with the mapped values
    bound_qc = QuantumCircuit(1)
    bound_qc.rx(theta1_value, 0)  # θ1
    bound_qc.ry(theta2_value, 0)  # θ2
    bound_qc.rz(theta3_value, 0)  # θ3
    bound_qc.measure_all()
```

```
Confusion Matrix:
[[ 4  2]
 [14 12]]
Precision (TP/(all predicted positives): 0.86
Recall (TP/ all actual positives)      : 0.46
Specificity TN/(all actual negatives)  : 0.67
Negative Predictive Value (TN/ All predicted negatives) (NPV): 0.22
Accuracy: 0.50
```

Project Phases Overview

Phase I
Classical Machine Learning

Phase II
Quantum Machine Learning Translation

Phase III
Naive Bayes in Classical and Quantum Methods

Phase I
Classical Machine Learning
(Predicting Game Winners)

Objective:

Build a classical machine learning model using Random Forest to predict game winners in the Premier League for Liverpool games in 2013-2014

Approach:

- Start with a Random Forest classifier to predict match outcomes (win, lose)
- Refine the model by domain knowledge, feature selection and adding more match features

TASKS

Data Preprocessing and Exploration

Gather and clean historical match data (e.g., goals, shots, possession, passes)

Feature Selection and Engineering

Select important match features such as home advantage, team form, player injuries, and head-to-head results

Model Training and Evaluation

Train the Random Forest model and evaluate its performance using metrics like accuracy, precision, and recall

Phase I - Classical Machine Learning

Task 1: Data Preprocessing and Exploration

- Prepare datasets and ensure the column names match for merging
- Rename columns to shorter but meaningful names and apply the renaming
- Select only the specified columns

```python
# Rename columns to shorter but meaningful names
columns_rename_map = {
    'Date': 'date',
    'home_team': 'home_team',
    'away_team': 'away_team',
    'FTHG': 'home_goals',
    'FTAG': 'away_goals',
    'FTR': 'ft_result',
    'HTHG': 'ht_home_goals',
    'HTAG': 'ht_away_goals',
    'HTR': 'ht_result',
    'Referee': 'referee',
    'HS': 'home_shots',
    'AS': 'away_shots',
    'HST': 'home_shots_on_target',
    'AST': 'away_shots_on_target',
    'HF': 'home_fouls',
    'AF': 'away_fouls',
    'HC': 'home_corners',
    'AC': 'away_corners',
    'HY': 'home_yellow_cards',
    'AY': 'away_yellow_cards',
    'HR': 'home_red_cards',
    'AR': 'away_red_cards'
}
```

Phase I - Classical Machine Learning

Task 2: Feature Selection and Engineering

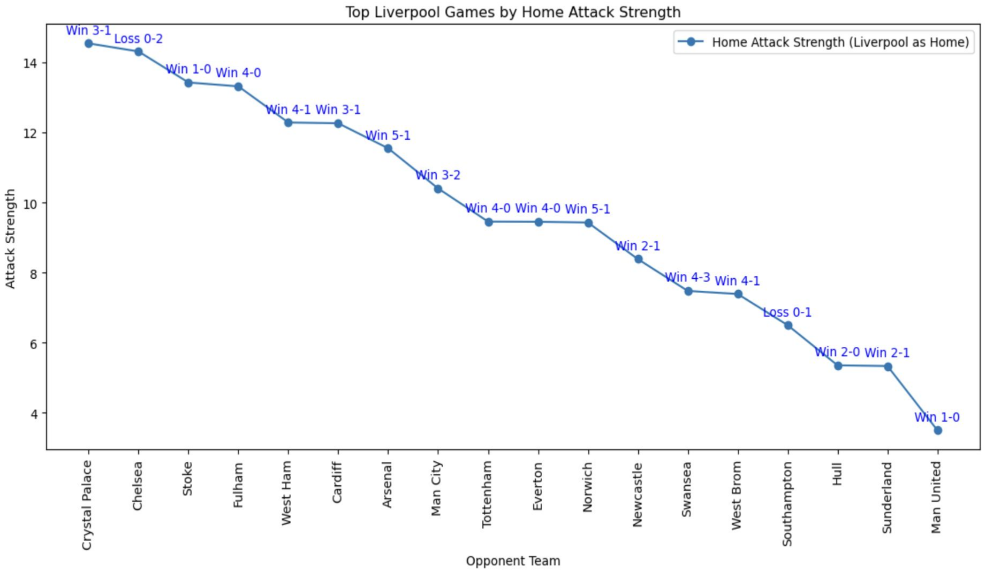
Home Attack Strength

- The number of goals the home team scores
- The shot on target / the total shots

  → How accurate the team's attacks are

- The number of corners

  → Opportunities to create chance

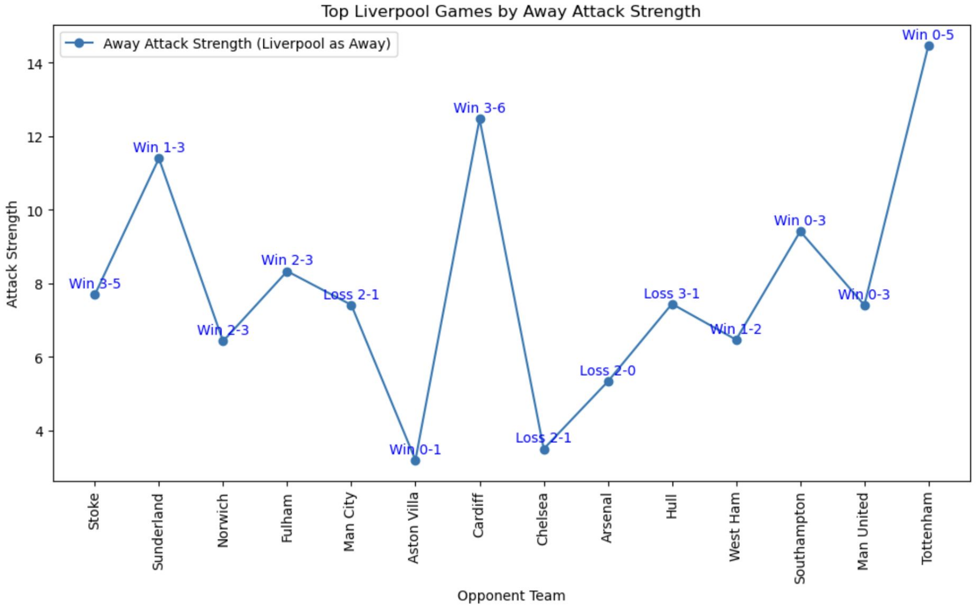


Phase I - Classical Machine Learning

Task 2: Feature Selection and Engineering

Away Attack Strength

- The goals scored by the away team
- The shot on target / the total shots
- The number of corners they win



Phase I - Classical Machine Learning

Task 2: Feature Selection and Engineering

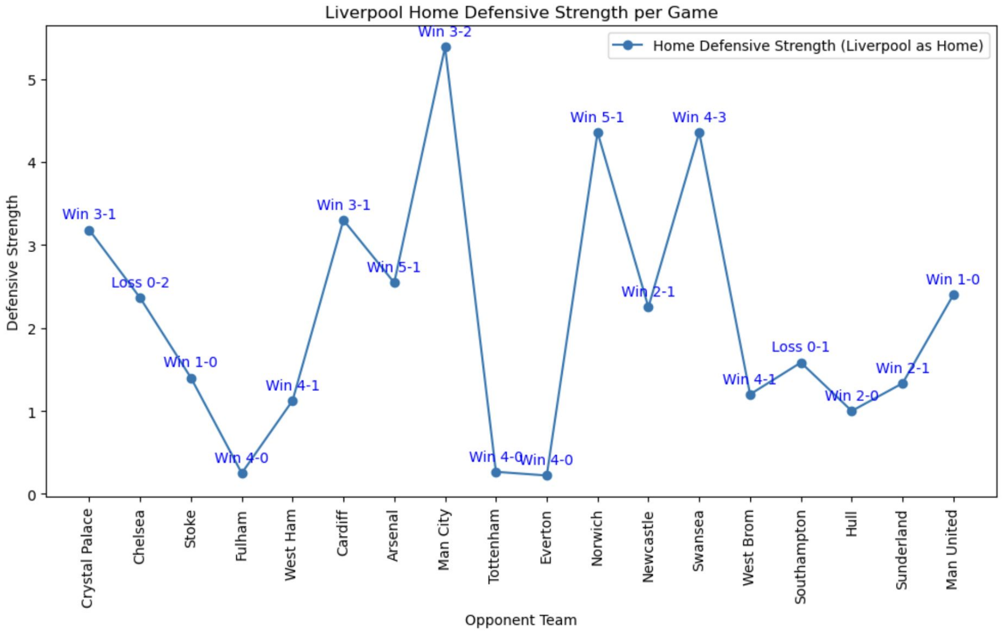
Home Defensive Strength

- The goals scored by the away team
- The shots on target they allow / the total shots from the away team
- The number of yellow cards they receive

  → More yellow cards suggest defensive mistakes

- The number of red cards

  → Red cards have double the impact

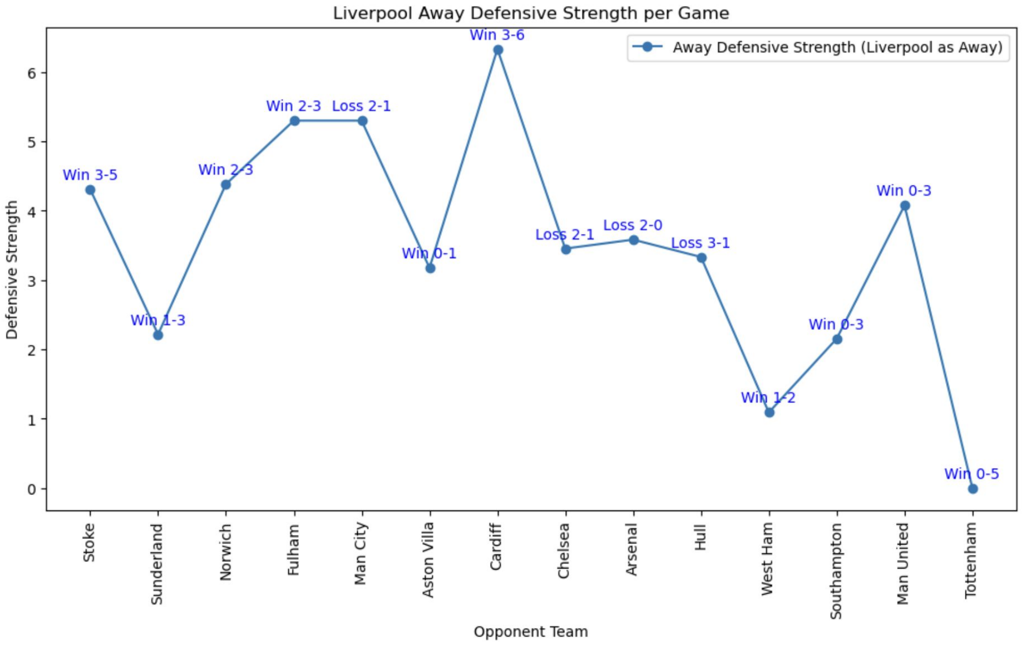


Phase I - Classical Machine Learning

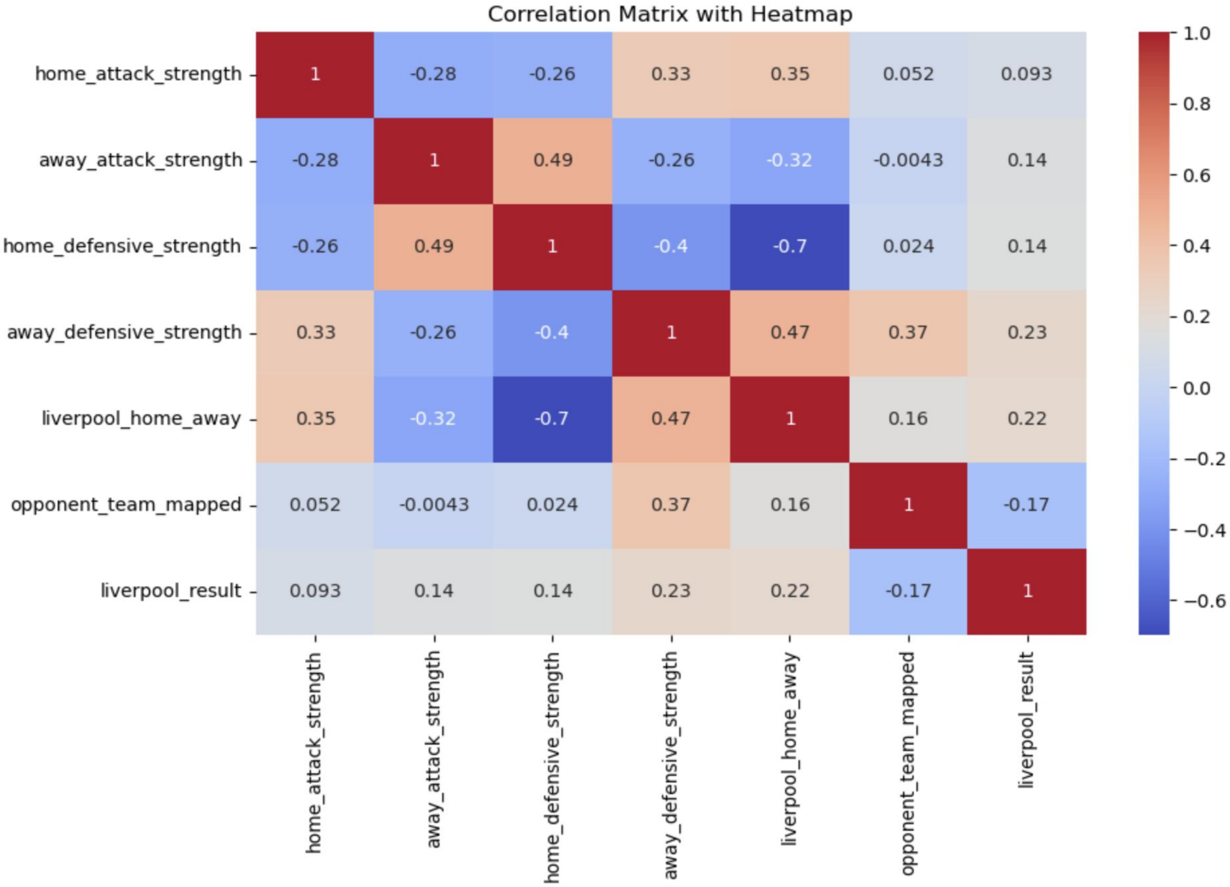
Task 2: Feature Selection and Engineering

Away Defensive Strength

- The goals scored by the home team
- The shots on target allowed by the away team / the total shots from the home team
- The number of yellow cards they receive
- The number of red cards



Correlation matrix (Heatmap)



Phase I - Classical Machine Learning

Task 2: Feature Selection and Engineering

- Filter matches where Liverpool is either the home team or away team
- Remove draw matches based on home and away goals
- Define a function to check if Liverpool won, then assign 1 for Liverpool won and 0 for Liverpool lost

```python
# Define a function to check if Liverpool won
def check_liverpool_result(row):
    if row['home_goals'] > row['away_goals'] and row['home_team'] == 'Liverpool':
        return 1  # Liverpool won at home
    elif row['away_goals'] > row['home_goals'] and row['away_team'] == 'Liverpool':
        return 1  # Liverpool won away
    else:
        return 0  # Liverpool lost
```

- Create a new column 'liverpool_home_away' where 1 is for home and 0 is for away
- Map opponent teams to the indices ranging from 2 to 20, while 0 is for Liverpool away and 1 is for Liverpool

```
{'Stoke City': 2,
 'Manchester United': 3,
 'Southampton': 4,
 'Crystal Palace': 5,
 'West Bromwich Albion': 6,
 'Fulham': 7,
 'Norwich City': 8,
 'West Ham United': 9,
 'Cardiff City': 10,
 'Hull City': 11,
 'Everton': 12,
 'Arsenal': 13,
 'Swansea City': 14,
 'Sunderland': 15,
 'Tottenham Hotspur': 16,
 'Manchester City': 17,
 'Chelsea': 18,
 'Newcastle United': 19,
 'Aston Villa': 20}
```

Phase I - Classical Machine Learning

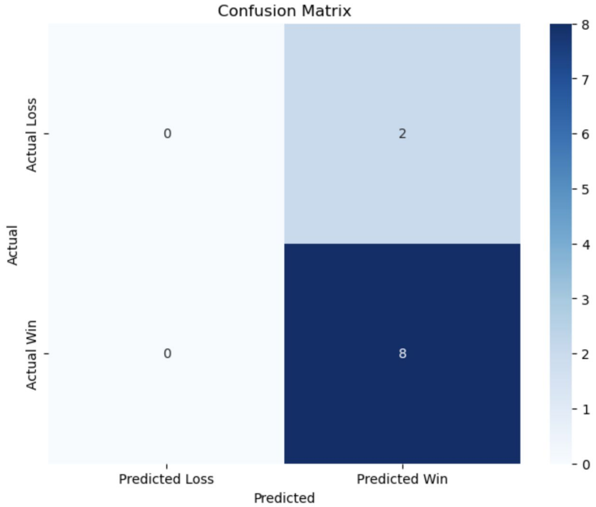
Task 3: Model Training and Evaluation

- Define feature set and labels

```
X = liverpool_matches_no_draws[['home_attack_strength', 'away_attack_strength', 'home_defensive_strength',
                               'away_defensive_strength', 'liverpool_home_away', 'opponent_team_mapped']]
y = liverpool_matches_no_draws['liverpool_result']
```

- Split the data into training and test sets
- Initialize Random Forest classifier
- Predict the results on the test set
- Calculate metrics for Random Forest

```
Confusion Matrix:
[[0 2]
 [0 8]]
Precision (TP/(all predicted positives): 0.80
Recall (TP/ all actual positives)      : 1.00
Specificity TN/(all actual negatives)  : 0.00
Negative Predictive Value (TN/ All predicted negatives) (NPV): 0.00
Accuracy: 0.80
```



Phase I - Classical Machine Learning

Task 3: Model Training and Evaluation

- Predict the results on the entire dataset

	Home/Away	Opponent	Real Result	Predicted Result	Correct Prediction	Real Points	Predicted Points
1	Yes	Stoke	Win	Win	True	3	3
11	No	Aston Villa	Win	Win	True	6	6
28	Yes	Man United	Win	Win	True	9	9
41	Yes	Southampton	Lose	Win	False	9	12
58	No	Sunderland	Win	Win	True	12	15
63	Yes	Crystal Palace	Win	Win	True	15	18
82	Yes	West Brom	Win	Win	True	18	21
90	No	Arsenal	Lose	Lose	True	18	21
103	Yes	Fulham	Win	Win	True	21	24
127	No	Hull	Lose	Win	False	21	27
133	Yes	Norwich	Win	Win	True	24	30
141	Yes	West Ham	Win	Win	True	27	33
159	No	Tottenham	Win	Win	True	30	36
162	Yes	Cardiff	Win	Win	True	33	39

Phase II
Quantum Machine Learning Translation

Objective:

Translate the classical model into a quantum machine learning model

Approach:

- Use parameterized quantum circuits (PQCs) to implement the model

Tasks:

- Translate classical features into quantum features
- Implement and test quantum model
- Evaluate the model

Overview of Steps

- **Quantum Circuit Construction:** Build a quantum circuit where the classical features are encoded into the quantum state via parameterized rotation gates
- **Execution on a Quantum Simulator:** Execute on the QASM simulator (which mimics the behavior of a quantum processor)
    - The circuit is run multiple times to gather measurement statistics.
- **Classification Based on Measurements:** Classify the match result (win or loss) based on the measurement result
    - If the qubit measurement is more likely to be 0, we predict a loss; if it's 1, we predict a win.

```python
# Define the quantum parameters
theta1 = Parameter('θ1')
theta2 = Parameter('θ2')
theta3 = Parameter('θ3')
theta4 = Parameter('θ4')
theta5 = Parameter('θ5')
theta6 = Parameter('θ6')

# Create a quantum circuit with one qubit
qc = QuantumCircuit(1)

# Add parameterized gates to the quantum circuit
# Rotate the qubit around the X-axis by theta1
qc.rx(theta1, 0)
# Rotate the qubit around the Y-axis by theta2
qc.ry(theta2, 0)
# Rotate the qubit around the Z-axis by theta3
qc.rz(theta3, 0)
# Rotate the qubit around the X-axis by theta4
qc.rx(theta4, 0)
# Rotate the qubit around the Y-axis by theta5
qc.ry(theta5, 0)
# Rotate the qubit around the Z-axis by theta6
qc.rz(theta6, 0)

# Add a measurement to the quantum circuit
qc.measure_all()
```

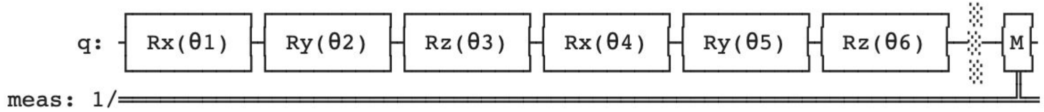
Phase II - One Qubit

Task 1: Translation from Classical to Quantum Features

- Map normalized features to a specific quantum gate parameter (θ1, θ2, etc.)
  - θ1 is derived from the normalized home attack strength.
  - θ2 is derived from the normalized away attack strength.
  - θ3 is derived from the normalized home defensive strength.
  - θ4 is derived from the normalized away defensive strength.
  - θ5 is derived from the home/away status.
  - θ6 is derived from the normalized opponent team mapping.

```
# Map input features to theta values with adjusted scaling factors
# θ1 is based on normalized home attack strength, scaled by π
theta1_value = home_attack_strength_normalized * np.pi

# θ2 is based on normalized away attack strength, scaled by π
theta2_value = away_attack_strength_normalized * np.pi

# θ3 is based on normalized home defensive strength, scaled by π
theta3_value = home_defensive_strength_normalized * np.pi

# θ4 is based on normalized away defensive strength, scaled by π
theta4_value = away_defensive_strength_normalized * np.pi

# θ5 is based on home/away status, scaled by π/2
theta5_value = home_away_normalized * np.pi / 2

# θ6 is based on normalized opponent team mapping, scaled by 2π/3
theta6_value = opponent_team_mapped_normalized * 2 * np.pi / 3
```

- Build a quantum circuit where the classical features are encoded into the quantum state via parameterized rotation gates




```
# Classify based on the measurement result
if counts.get('0', 0) > counts.get('1', 0):
    classification = 0  # Predicts loss
else:
    classification = 1  # Predicts win

return classification, label
```

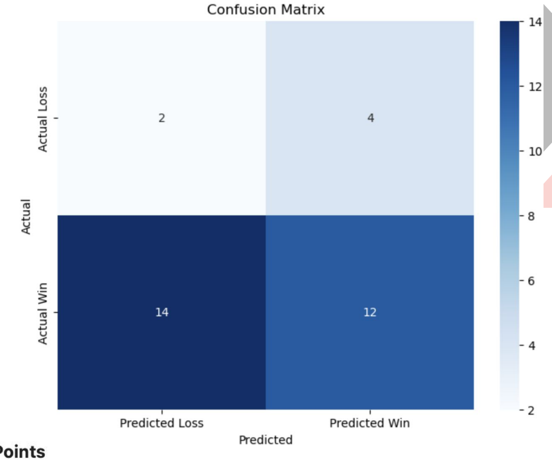
Task 2: Quantum Model Implementation and Testing

- Execute it on the QASM simulator, which mimics the behavior of a quantum processor
- Classify the match result (win or loss) based on the measurement result
  - If the qubit measurement is more likely to be 0, we predict a loss; if it's 1, we predict a win

Phase II - One Qubit

Task 3: Model Evaluation & Prediction

```
Confusion Matrix:
[[ 2  4]
 [14 12]]
Precision (TP/(all predicted positives): 0.75
Recall (TP/ all actual positives)       : 0.46
Specificity TN/(all actual negatives)   : 0.33
Negative Predictive Value (TN/ All predicted negatives) (NPV): 0.12
Accuracy: 0.44
```



- Predict the results

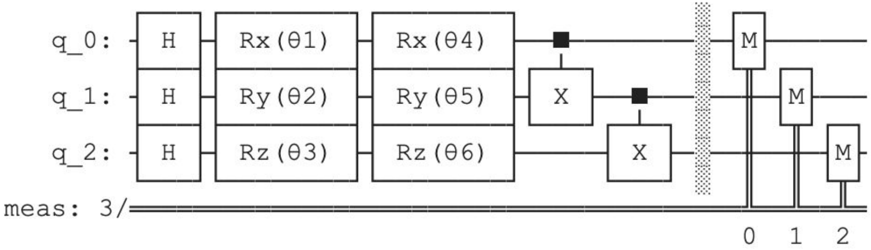
	Home/Away	Opponent	Real Result	Predicted Result	Correct Prediction	Real Points	Predicted Points
1	Yes	Stoke	Win	Lose	False	3	0
11	No	Aston Villa	Win	Win	True	6	3
28	Yes	Man United	Win	Win	True	9	6
41	Yes	Southampton	Lose	Win	False	9	9
58	No	Sunderland	Win	Lose	False	12	9
63	Yes	Crystal Palace	Win	Lose	False	15	9
82	Yes	West Brom	Win	Win	True	18	12
90	No	Arsenal	Lose	Win	False	18	15
103	Yes	Fulham	Win	Lose	False	21	15
127	No	Hull	Lose	Win	False	21	18
133	Yes	Norwich	Win	Win	True	24	21
141	Yes	West Ham	Win	Lose	False	27	21
159	No	Tottenham	Win	Win	True	30	24
162	Yes	Cardiff	Win	Lose	False	33	24

Phase II - Multiple Qubits

Task 1: Multiple Qubits Preparation

- Define the quantum parameters
- Create a quantum circuit with multiple qubits
- Add Hadamard gates to create superposition
- Add paratermized gates to the quantum circuit
- Add CNOT gates to create entanglement
- Add a measurement to a quantum circuit

```
# Create a quantum circuit with multiple qubits
qc = QuantumCircuit(3)
```



Task 2: Same Tasks Applied to Multiple Qubits

- Normalize the input features to a range [0, 1]
- Map input features to theta values with adjusted scaling factors
- Create a new quantum circuit with mapped values
- Execute the circuit on qasm_simulator for measurement
- Classify based on measurement result

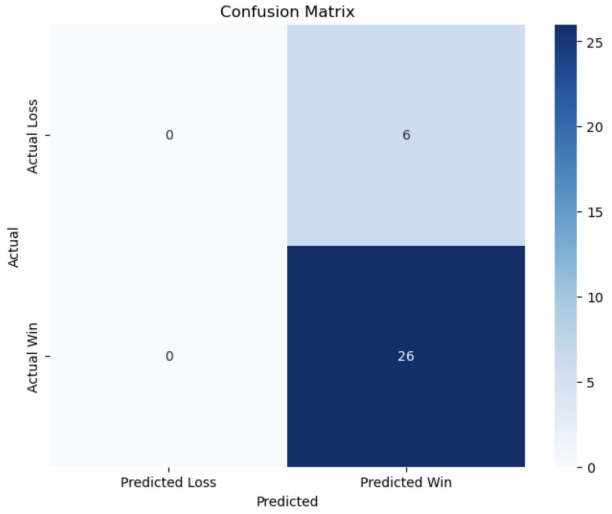
```
Classification results: [(1, 1), (1, 1), (1, 1), (1, 0), (1, 1), (1, 1), (1, 1), (1, 0), (1, 1), (1, 0), (1, 1), (1, 1), (1, 1), (1, 1), (1,
0), (1, 0), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 0), (1, 1)]
Actual labels: (1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1)
```

Phase II - Multiple Qubits

Task 3: Model Evaluation & Result Prediction

```
Confusion Matrix:
[[ 0  6]
 [ 0 26]]
Precision (TP/(all predicted positives): 0.81
Recall (TP/ all actual positives)       : 1.00
Specificity TN/(all actual negatives)   : 0.00
Negative Predictive Value (TN/ All predicted negatives) (NPV): nan
Accuracy: 0.81
```

	Home/Away	Opponent	Real Result	Predicted Result	Correct Prediction	Real Points	Predicted Points
1	Yes	Stoke	Win	Win	True	3	3
11	No	Aston Villa	Win	Win	True	6	6
28	Yes	Man United	Win	Win	True	9	9
41	Yes	Southampton	Lose	Win	False	9	12
58	No	Sunderland	Win	Win	True	12	15
63	Yes	Crystal Palace	Win	Win	True	15	18
82	Yes	West Brom	Win	Win	True	18	21
90	No	Arsenal	Lose	Win	False	18	24
103	Yes	Fulham	Win	Win	True	21	27
127	No	Hull	Lose	Win	False	21	30
133	Yes	Norwich	Win	Win	True	24	33
141	Yes	West Ham	Win	Win	True	27	36
159	No	Tottenham	Win	Win	True	30	39
162	Yes	Cardiff	Win	Win	True	33	42



PQC Enhancements for Predictive Modeling

Quantum Superposition

Represent multiple states simultaneously, allowing for more efficient exploration of the solution space

Quantum Entanglement

Capture complex relationships between features, making it more adaptable to different kinds of data

Improvement over Classical

Higher accuracy, faster training for larger dataset

Benefits of PQCs in Football Analytics

Enhanced Accuracy

Recognize complex patterns and interactions between features, leading to more precise predictions and insights

Scalability

Process vast amounts of football data, such as player performance metrics, team foundations, and match events, handling larger datasets efficiently

Speed

Provide faster training times enabling real-time analysis, which is crucial for in-game strategies and post-match evaluations

Conclusion

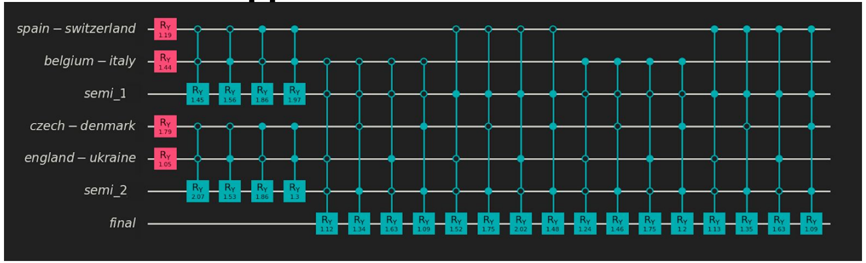
	Quiz 4 Classical	Quiz 4 Quantum	Classical	One Qubit Quantum	Multiple Qubits Quantum	Naive Bayes
Accuracy Score	0.70	0.62	0.80	0.44	0.81	0.80

Future Directions:

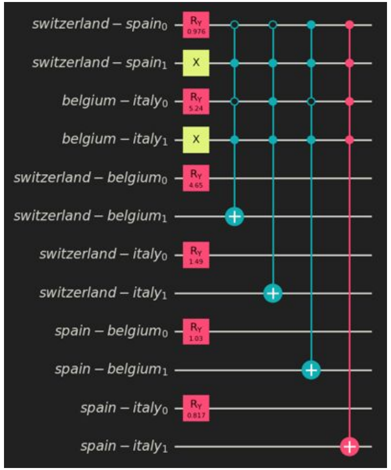
- Real-time data integration, including data from matches, player statistics, and external factors
- Apply PQCs to other sports to gain insights into player performance, team dynamics and match outcomes across different context
- Apply hybrid classical-quantum models, using classical algorithms for initial data processing and quantum techniques for complex pattern recognition and optimization

Reference - How Qubits Can Predict Euro 2020

- **Base assumption:** The winning probability for each match between two specific teams is predetermined and known.
  - A probability p for the first team to win, and probability (1-p) for the second team to win
  - $0 \leq p \leq 1$
- **Round-based Approach**



- **Matchup-based Approach:** More qubits, fewer gates



Source Link : https://www.classiq.io/docs/quantum-football-how-qubits-can-predict-euro-2020

Questions & Answers

Thank you for your attention! Any questions?