NFL Game Predictions

Group 6 : Thomas Hervé Muel [1], Benjamin Kahkeshan[2]

[1]muel@kth.se

[2]benkah@kth.se

Abstract

Predicting the results of sports competitions is one of the most fascinating and challenging topics in AI nowadays because it has always been associated with numerous complexities. To predict the games there are a lot of parameters to consider. The National Football League (NFL) games are no exception to this rule. These factors include the current condition of the teams, player injuries and weather conditions. Despite these complexities the NFL provides extensive and accurate statistical data that are very important for predicting game outcomes.

In this project, historical NFL team statistics are used to predict home-team victories using machine learning models and especially Neural Networks. Team statistics such as offense, defense and scoring are combined and the differences between the home and away teams are used as input features. This data is used to evaluate three models: a classical neural network, a quantum neural network and a hybrid classical–quantum model. PCA is applied to reduce feature dimensionality where needed.

In external evaluation on 31 played upcoming games, the classical model accuracy is 64.52% (20/31), hybrid model accuracy is 61.29% (19/31), quantum model accuracy is 51.61% (16/31), classical PCA model accuracy is 58.06% (18/31) and the hybrid PCA model accuracy is 64.52% (20/31). While the results are deceiving they really show that predicting NFL games are hard even if the models showed good accuracies on the testing dataset. Our classical models had 80% accuracy. Our hybrid models had 73.91% without PCA and 82.61% accuracy with. The quantum model has shown the worst results with 69.57% accuracy. While we are only talking about the testing metrics now and not the training and validation we are going to talk about them later.

Keywords: NFL Prediction, Machine Learning, Quantum Computing, Variational Quantum Classifier (VQC), Hybrid Neural Networks, Sports Analytics, Binary Classification.

1. Introduction

Advances in digital technologies have led to the collection of large volumes of structured sports data. The National Football League (NFL) also collects, organizes and makes available extensive statistical data to enthusiasts and researchers across multiple seasons. This increase in the volume of structured data in American football has enabled the effective use of machine learning and neural network models to analyze team performance and predict match results.
For this reason, this project aims to investigate and compare different Neural Network based approaches for predicting NFL game outcomes and focuses on evaluating the performance of classical neural networks, quantum machine learning models and hybrid classical–quantum architectures.

1.1 Background and Related Work

In sports, predicting outcomes and extracting information is important for professionals and the general public, especially in team management and betting. The increasing volume of available data has led to the widespread use of data mining techniques and machine learning algorithms to predict outcomes based on input data. Previous studies indicate that classification-based models perform better than regression models. Feature selection and feature creation are critical in increasing the accuracy and interpretability of the models [1].

In professional leagues such as American football, due to the high popularity of this sport, a large amount of structured statistical data about games and team performance is available. This data which is collected through publicly available NFL statistical databases, includes offensive and defensive indicators of teams. In the preprocessing stages, features such as offensive statistics and defensive indicators are usually used. Since each match involves direct competition between two teams, in many studies the difference between the home and away team performance is considered as input to the model in order to make a more accurate comparison between the teams [2].

One common requirement before feeding datasets to machine learning models is to standardize and scale the features, as many algorithms do not perform well when these features have different scales. however, high-dimensional datasets are usually difficult to interpret which makes the use of methods such as Principal Component Analysis (PCA) important. PCA reduces the dimensionality of the data, making it easier to interpret while losing the minimum possible amount of information [3].

Several studies have shown that machine learning models can perform well in predicting the results of sports competitions. Methods such as logistic regression, decision trees, random forests and neural networks have been used in various studies to analyze sports data and predict match outcomes [2][4]. Neural networks, due to their ability to model nonlinear relationships, are considered particularly suitable for this type of problem.

Recently, some studies have investigated the application of quantum machine learning to binary classification problems [5][6]. These studies have tried to study the potential capabilities of quantum circuits compared to classical methods, although due to current technological limitations, most of these studies have been conducted on small-scale datasets or using quantum simulators.

Furthermore, hybrid classical–quantum models have been introduced in recent studies [7]. In these approaches, classical neural networks are responsible for feature extraction and the quantum component is used to perform the final classification step. These architectures are proposed as a practical solution for investigating quantum machine learning on real-world applications.
The methodology of this project is based on the implementation and evaluation of several machine learning approaches.

1.2 Problems and Aim

Predicting NFL game outcomes using team statistics is a binary classification problem. Our goal in this project is to estimate the outcome of a game before it happens with high predictive accuracy. In other words we are trying to find the best neural network architecture to predict the NFL games outcomes. In various studies, classical machine learning models in sports data analysis have predicted outcomes with acceptable accuracy [8][1], but it is still unclear to what extent emerging technologies such as quantum machine learning models or hybrid classical–quantum models can provide acceptable performance for this type of problem. This is also what we are trying to investigate.

1.3 Problem Statement

Given the team statistics of two NFL teams before a game, how accurately can we predict whether the home team will win/lose the game?

To answer this question we were aiming for at least 60% of the predictions to be correct. We designed a dataset in which the features represent the differences between the home and away teams' performance in offensive, defensive, scoring and other related statistics. We implemented five machine learning models following 3 architectures: classical neural network, quantum classification and hybrid classical-quantum. Using a shared dataset we evaluate and compare their efficiency in predicting NFL game outcomes. Furthermore we compare these models in terms of prediction accuracy and computational performance, including evaluation with and without PCA.

- Build a repeatable dataset by merging NFL statistics and difference-based features.
- Implement a classical neural network.
- Develop a quantum neural network using a variational quantum circuit.
- Create a hybrid model combining quantum circuits with classical neural network layers.
- Test the impact of PCA on model performance.
- Evaluate all models on historical data and unseen future games.
- Analyze the practical advantages and current limitations of quantum approaches in sports prediction.

Based on existing work in sports outcome prediction, these objectives define the project's workflow, starting with data preparation and moving through model implementation, testing and final analysis.

2. Contribution

The goal of this project was to compare classical, quantum and hybrid machine learning models for predicting NFL game outcomes using the same data and evaluation process. The results show that classical and hybrid models currently outperform the purely quantum model, highlighting both the limitations and potential of quantum approaches. This work is mainly relevant for students and researchers in machine learning, quantum computing and sports analytics. This project also contributes to the growing body of empirical benchmarks for Quantum Machine Learning, providing rare real-world performance data on tabular datasets compared to standard toy datasets (like MNIST). Ethical concerns may arise if these models are used for gambling or important decisions, and quantum simulations may also raise sustainability concerns because they require a lot of computing power.

2.1 Design

The system uses one shared pipeline so all models are compared fairly with the same inputs and the same target label. The inputs are team statistics and the game schedule (home vs visitor). For each game, we merge the home team stats and the visitor team stats into a common pandas list. Next, we build the main features by taking the difference between the two teams for every feature seen in the databases.

This turns each game into one feature vector that shows how the home team compares to the visitor. Before training, we standardize the features using StandardScaler (z-score) to make the training faster. PCA can also be used as an optional step to reduce the number of features. Finally, the same processed features are given to three model types (classical DNN, quantum VQC, and hybrid). All models output the same result: the probability that the home team wins, P(Home win).

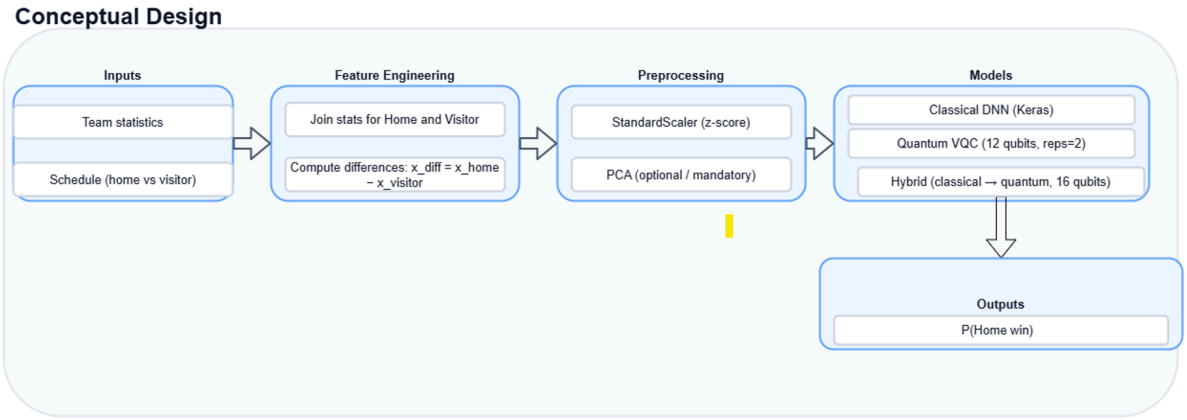

Figure 1: Data Flow and Conceptual Design of the Prediction Pipeline (Data Engineering to Model Output).

2.1.1 Requirements

Functional Requirements (What the system does):

1. The system shall scrape historical and upcoming game data from NFL websites.
2. The system shall preprocess data to calculate difference vectors (Home - Visitor).
3. The system must support three distinct architectures: Classical, Quantum (VQC), and Hybrid.
4. The system must output a binary classification (Win/Loss) and a probability score.

Non-Functional Requirements (How the system performs):

1. Accuracy**:** The model shall achieve >60% accuracy on unseen external data.
2. Reproducibility**:** All data splits and model initializations must use fixed random seeds.
3. Performance**:** The inference time for predictions must be reasonable for batch processing (under 1 minute for the full schedule).

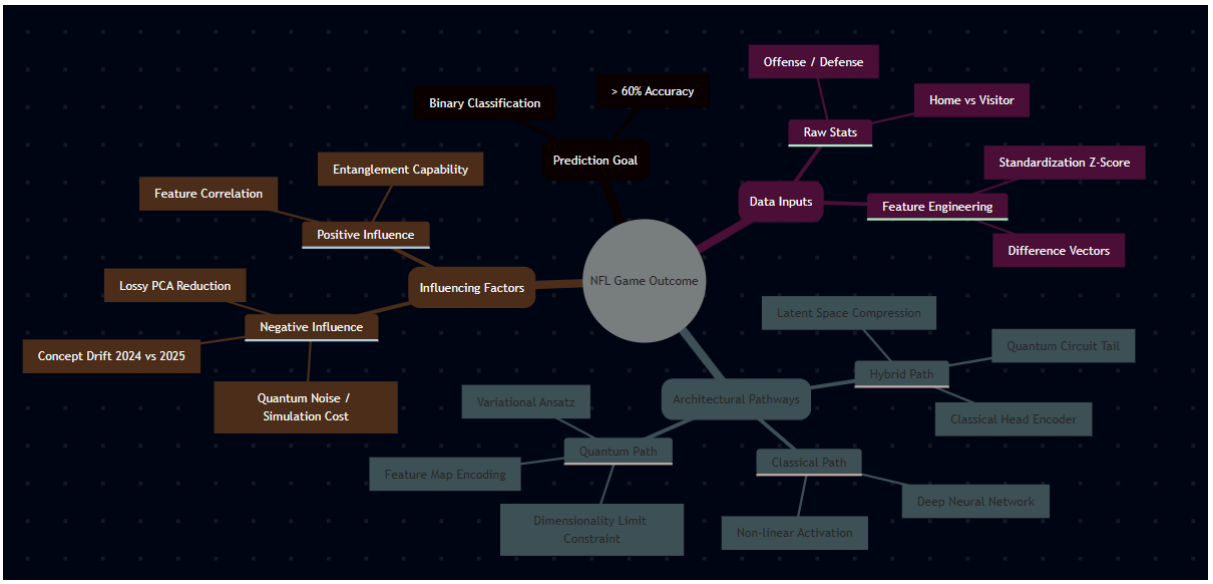

Figure 2: Cognitive Map of the Prediction System. This diagram illustrates the causal relationships between data engineering concepts, architectural choices (Classical vs. Quantum), and the external factors (noise, drift) that influence the final prediction accuracy.

2.1.2 Deliverables

The primary deliverables of this project are:

- A Python-based Data Pipeline: Scripts (get_data.py, data_processing.py) for scraping and cleaning NFL statistics.
- Three Predictive Models: Implemented architectures for Classical (Keras), Quantum (PennyLane), and Hybrid (PyTorch) classifiers.
- User Guide: Documentation for reproducing the environment and executing predictions.

2.2 Architecture

The system is modular and implemented as separate scripts with clear inputs/outputs. Raw tables are stored in prep_data/, processed model inputs are stored in processed_data/ and model predictions are exported to predictions/. The workflow is:

get_data.py collects and stores raw statistics and schedules in *prep_data/*

data_processing.py merges tables and produces processed datasets in *processed_data/*

neural_network.py, quantum_neural_network.py and hybrid_neural_network.py train models and export prediction files to *predictions/*

evaluate_models.py merges predictions with played-game results and computes external accuracy

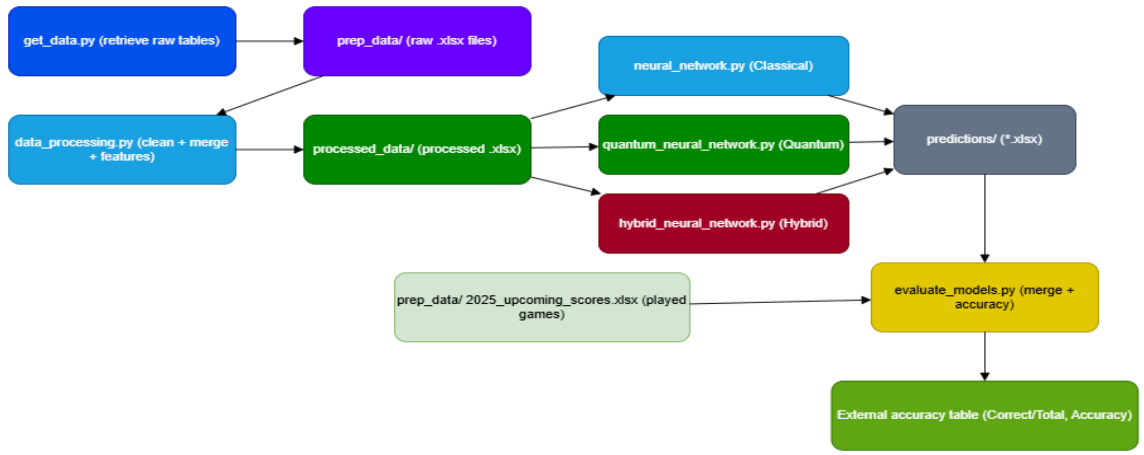

Figure 3: Project Implementation Architecture and Script Workflow.

2.3 Results and Evaluation

2.3.1 Results

External and Internal evaluation (played upcoming games).

External accuracy is computed by merging each model's prediction file with the played upcoming-games file and calculating accuracy on the matched subset only. In this evaluation, 31 games matched across datasets.

Internal accuracy corresponds to how well our models performed on the test dataset.

Table 1: Comparative Results of Classical, Hybrid, and Quantum Models on Internal and External Datasets.

Model	Correct / Total (External)	Accuracy (External)	Correct / Total (Internal)	Accuracy (Internal)
Hybrid	19/31	60.29%	17/23	73.91%
Classical	20/31	64.52%	36/45	80%
Quantum	16/31	51.61%	16/23	69.57%
Classical + PCA	18/31	58.06%	36/45	80%
Hybrid + PCA	20/31	64.52%	19/23	82.61%

Overall for the upcoming games predictions, Hybrid with PCA and Classical achieved the highest external accuracy (64.52%) while the Quantum-only model achieved the lowest (51.61%). On the internal testing (while creating the model) we see that the hybrid + PCA is showing the best results (82.61%) and the quantum is showing the worst (69.57%). It reflects what we have seen on the future predictions but the accuracy of the predictions has dropped significantly.

2.3.2 Testing

Each model is evaluated on a held-out test split using a confusion matrix and a classification report (precision, recall and F1-score). Training curves (training loss and accuracy and validation loss and accuracy) are inspected to assess convergence and potential overfitting.

In the table below 0 corresponds to the games lost by the home team and 1 the games won by the home team.

Table 2: Detailed Classification Report (Precision, Recall, F1-Score) and training time on Test Data.

Model	Precision (0)	Recall (0)	F1-score (0)	Precision (1)	Recall (1)	F1-score (1)	Training Time (s)
Classical	86.67%	65.00%	74.29%	76.67%	92.00%	83.64%	2.39
Classical + PCA	72.00%	90.00%	80.00%	90.00%	72.00%	80.00%	2.93
Hybrid	75.00%	60.00%	66.67%	73.33%	84.62%	78.57%	6.34
Hybrid + PCA	80.00%	80.00%	80.00%	84.62%	84.62%	84.62%	7.67
Quantum	66.67%	60.00%	63.16%	71.43%	76.92%	74.07%	10.98

We have also decided to show you the different performances of our models.

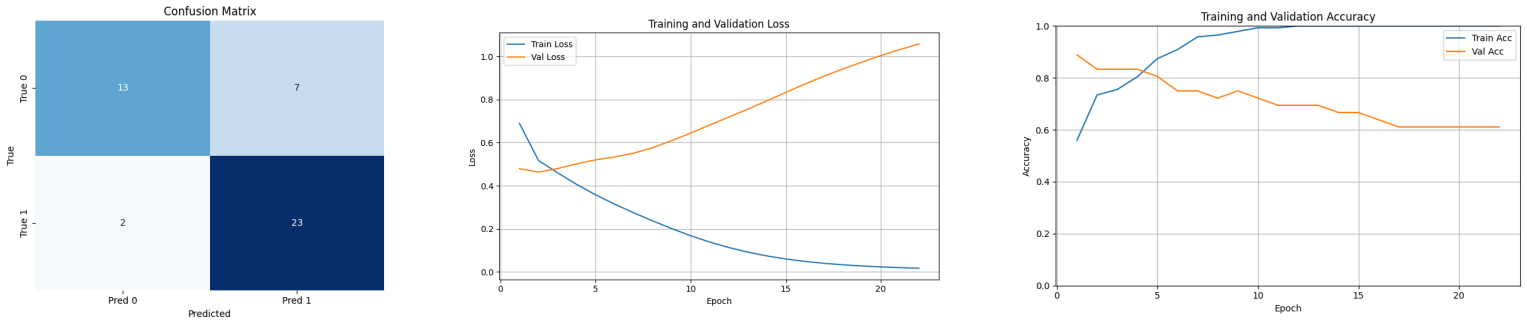

Figure 4: Classical Model Performance: Confusion Matrix and Training Curves.

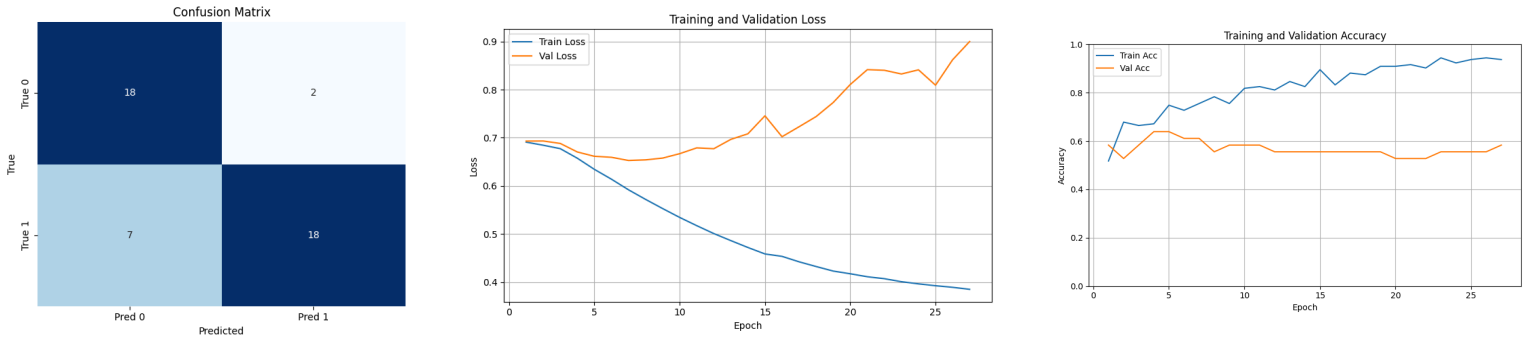

Figure 5: Classical Model with PCA: Confusion Matrix and Training Curves

Figure 6: Hybrid Model Performance: Confusion Matrix and Training Curves.

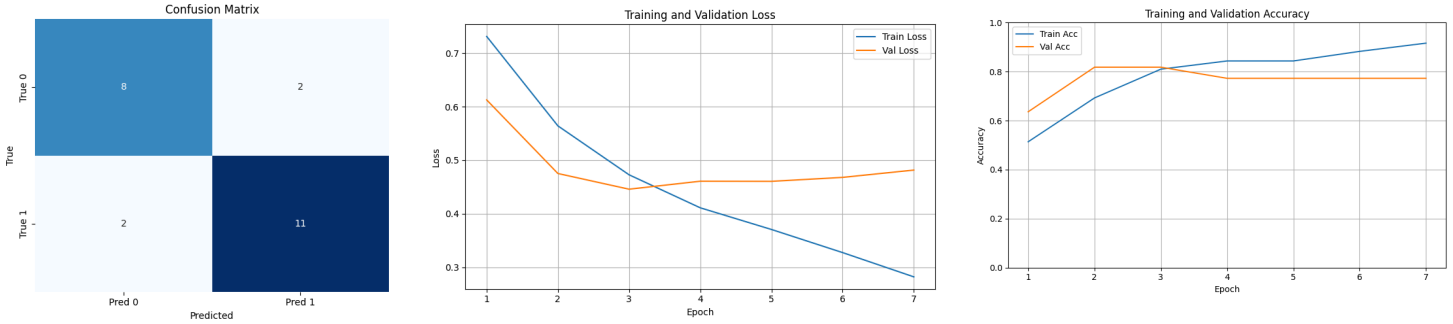

Figure 7: Hybrid Model Performance: Confusion Matrix and Training Curves.

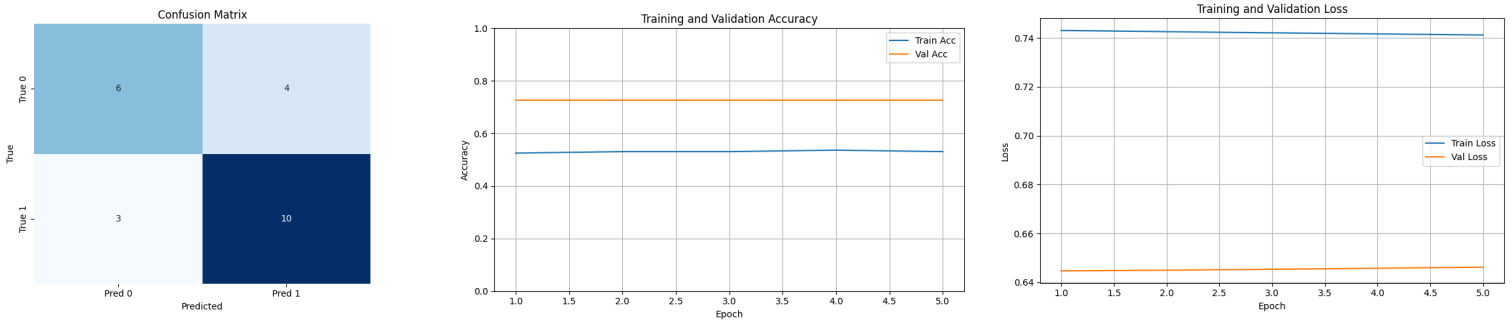

Figure 8: Quantum VQC Model Performance: Confusion Matrix and Training Curves.

2.3.3 Evaluation

The evaluation of the models was conducted on two distinct datasets: an internal "held-out" test set derived from historical data (Section 2.3.2) and an external dataset of 31 upcoming games played after the model training phase (Section 2.3.1).

Internal vs. External Performance: A significant discrepancy was observed between internal validation metrics and external real-world performance.

- Classical Model: Achieved 80% accuracy internally but dropped to 64.52% on external games.
- Hybrid Model (with PCA): Showed the strongest internal performance at 82.61%, yet also dropped to 64.52% externally.
- Quantum Model: Consistently underperformed, achieving 69.57% internally and only 51.61% externally, barely better than random chance.

This "generalization gap" suggests that while the Classical and Hybrid models successfully learned patterns in the historical training data (evident from the high ~80% internal accuracy), they suffered from "concept drift" where the statistical profile of the upcoming 2025 games differed slightly from the historical average. We have prevented the overfitting by applying early stopping and rewind to the best model. We also see that the validation accuracy of almost every model stays close to 80% which shows that it’s not only based on luck that we have good results.

The Hybrid approach (specifically with PCA) demonstrated that quantum circuits can function effectively when supported by classical feature processing. Using the properties of entanglement and superposition it has been able to discover some patterns between the input features and the game prediction. By reducing the dimensionality before the quantum stage, the Hybrid model achieved the highest internal accuracy (82.61%), outperforming the pure Classical model (80%). However, the Pure Quantum (VQC) model struggled significantly. This is likely due to the "curse of dimensionality" and the limited expressivity of a 7-qubit ansatz, which may not be sufficient to capture the complex non-linear relationships in NFL statistics without the help of classical feature extraction.

Ultimately, the Classical and Hybrid models met the project's success criterion of >60% accuracy on external data, while the pure Quantum model failed to do so.

3 Implementation

This section presents the implementation of the NFL game outcome predictions system. The workflow is structured into three stages: data collection and preprocessing, model-specific training and a common evaluation procedure. These stages are implemented in separate scripts to ensure reproducibility and a consistent comparison between the classical, quantum and hybrid models.

3.1 Development Method (Prototype Workflow)

The project was developed using a prototype-style workflow. We first defined the requirements and evaluation criteria, then designed the data pipeline and system structure. After that we implemented preprocessing and the three model pipelines iteratively, testing each version on a separate test set and using external game results when available. Overall, the development follows:

Requirements → Design → Architecture → Development → Testing → Delivery

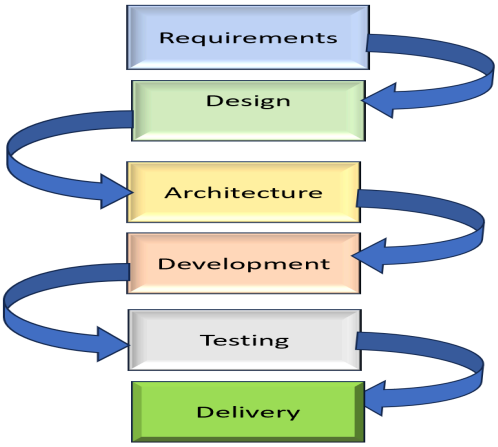

Figure 9: Agile Development Workflow.

As part of the final delivery, the system is designed to be executed via command-line interface. The user first runs python get_data.py --week [W] to scrape current data, followed by python data_processing.py to generate feature vectors. Finally, any of the three model scripts (e.g., python hybrid_neural_network.py) can be executed to generate the predictions/ Excel files. There will be more details in the READ_ME.md.

3.2 Data Collection and Preprocessing

Statistics are collected by category and side (offense/defense), including totals, passing, rushing, returns, punting, scoring and downs and saved as Excel files in prep_data/. Game schedules/results are also extracted and stored as separate datasets: games from weeks 1 to W (set via --week) are used as training scores while future matchups (weeks W+1 onward) are saved as upcoming games. In addition, an “upcoming scores” file keeps only upcoming games with dates up to yesterday (already played games), enabling external evaluation once results are available. Unfortunately we were unable to directly download the files from the website so we have implemented a script using the libraries requests, beautiful soup and pandas to scrap the website and get the data in a pandas list before saving them into an excel file. Some of the stats had to be clean especially for the team names in order for them to match the schedules and some data also needed to be rewritten in order for them to be considered as floats.

Before merging, statistics tables are cleaned by removing Gms columns and standardizing team names (e.g., trimming whitespace). All tables are then merged into a single team_data dataset and feature suffixes (e.g., _passing_offense, _scoring_defense) are added to preserve the source and avoid name collisions.

For each game, team_data is joined twice (home and visitor). Columns are renamed with _home and _visitor and difference-based features are computed as x_diff = x_home − x_visitor. The original _home/_visitor columns are removed so all models use the same _diff feature set. For historical games, Home_win is generated from the final score (Home_pts > Visitor_pts) and score columns are removed to prevent leakage. Final dataset are saved as:

processed_data/2025_processed_scores.xlsx
processed_data/2025_processed_upcoming_games.xlsx

3.3 Classical Model Pipeline (TensorFlow/Keras)

The classical model is implemented as a feed-forward neural network using TensorFlow/Keras. Input features are all columns ending with _diff, loaded from the processed dataset. The data is split into training and test sets (80/20) using a fixed random seed to also allow reproducibility. This split is not stratified and validation data is created internally during training using validation_split=0.2 from the training portion. Because the validation set is taken from the 80% training split, the effective data usage is approximately 64% for training, 16% for validation and 20% for testing.

All features are standardized using z-score normalization (StandardScaler), fitted on the training set and applied to the test set and upcoming games. PCA is implemented as an optional preprocessing step; when enabled, 32 components are used.

The classic neural network consists of Dense layers with ReLU activations and a single sigmoid output. Training uses Adam optimization and binary cross-entropy loss. Early stopping monitors validation loss (patience 20) and restores the best weights. Model performance is reported using a confusion matrix and classification report on the test set. Predictions for upcoming games are saved to Excel for later external evaluation.

Here is a visualization of the architectures of the neural network.

Figure 10: Classical Neural Network Architecture (Feed-Forward)

3.4 Quantum Model Pipeline (PennyLane + PyTorch)

The quantum model is implemented as a variational quantum classifier (VQC) using PennyLane with a PyTorch training loop. Due to the fixed circuit width, PCA is mandatory (because we runned the quantum circuit on a simulator and we need 1 qubit per feature. Having 1 more qubit/feature doubles the training time) : after standardization (StandardScaler), the feature vector is reduced to 7 principal components, matching the 7-qubit simulator (lightning.qubit).

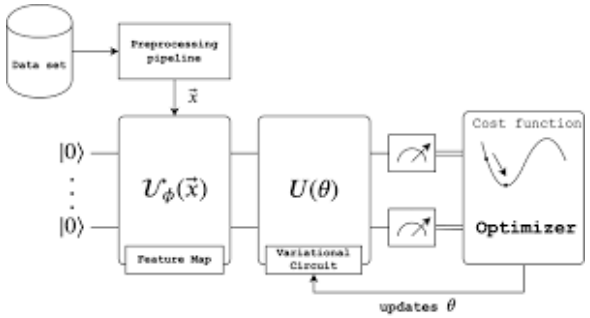

Figure 11 : Variational Quantum Classifier (VQC) Circuit Architecture. (Source : https://www.researchgate.net/publication/343063353_Dementia_Prediction_Applying_Variational_Quantum_Classifier/figures?lo=1&utm_source=google&utm_medium=organic)

Above is an image of the VQC architecture used. A Variational Quantum Classifier (VQC) is a hybrid algorithm that uses a parameterized quantum circuit to classify data. It begins with a feature map, which encodes classical input data into a quantum state, effectively mapping it into a high-dimensional Hilbert space. This state is then processed by a variational ansatz (a sequence of trainable gates and entanglers) whose parameters are iteratively optimized by a classical computer to minimize classification error.

The dataset is split into training, validation and test sets (80/10/10) using stratified sampling to preserve class balance. All tensors are converted to float precision (float32) for improved numerical stability and compatibility with the quantum differentiation pipeline.

Inputs are encoded using the AngleEmbedding feature map, followed by the Strongly Entangling Layers with reps = 2 ansatz. The circuit measures the expectation value of Pauli-Z on qubit 0 and maps it to a probability (since ⟨Z⟩ is between -1 and 1) using: p = (⟨Z⟩ + 1) / 2

Since the QNode does not natively process batches, a custom BatchedQNodeLayer iterates over samples in each mini-batch, evaluates the circuit per sample and stacks the outputs to enable mini-batch training with BCELoss and Adam. Early stopping is applied based on validation loss (patience = 5), restoring the best-performing model state. Because the quantum circuit is evaluated once per sample inside a Python loop, the quantum training pipeline is computationally heavier than the classical especially when using mini-batches. Here is the architecture saved :

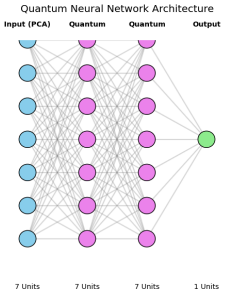

Figure 12 : Quantum Neural Network Architecture.

3.5 Hybrid Model Pipeline (PyTorch + PennyLane)

The hybrid architecture leverages the feature extraction capabilities of classical neural networks combined with a quantum classification layer.

The dataset is split using a stratified 80/10/10 strategy to preserve class balance. In this pipeline, the classical "head" acts as a trainable dimensionality reduction technique. It consists of three linear layers (128, 128, 4 units) using ReLU activations for the hidden layers and a Tanh activation (better for quantum) for the output. This maps the high-dimensional input features down to 4 parameters, which are scaled by **TT** to serve as rotation angles for the quantum circuit.

The quantum "tail" is a 4-qubit circuit using AngleEmbedding and StronglyEntanglingLayers (reps=3). Similar to the pure quantum model, the final classification is derived from the expectation value of the Pauli-Z operator on the zeroth qubit. This architecture allows the model to learn optimal features for the quantum circuit rather than relying solely on fixed mathematical reductions like PCA.

Here are the saved architectures :

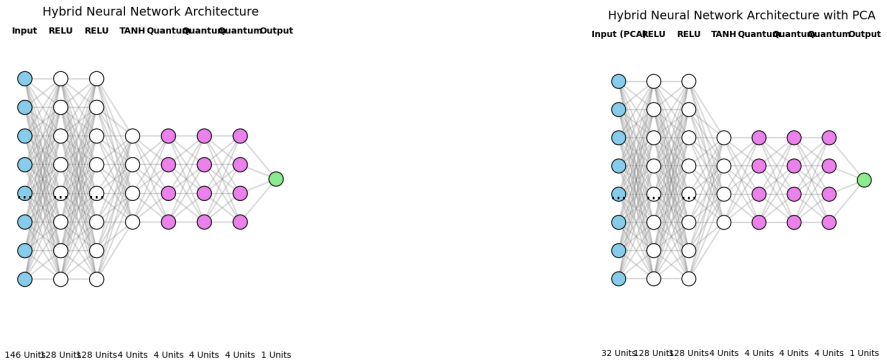

Figure 13 : Hybrid Quantum-Classical Neural Network Architecture.

3.6 Predictions

All the predictions were saved in a predictions folder containing the predictions for each model.

4. Conclusions and Discussion

4.1 Conclusions

The primary aim of this project was to investigate whether emerging quantum machine learning (QML) architectures could compete with or enhance classical methods in predicting NFL game outcomes. Our results indicate that while Quantum methods are promising, they are currently limited by hardware simulation constraints and feature mapping complexities.

We successfully met our problem statement goal of achieving >60% accuracy for the Classical (64.52%) and Hybrid (64.52%) models on unseen external data. The study concludes that Hybrid architectures represent the most viable path forward for QML in sports analytics today. They combine the robustness of classical feature extraction with the potential of quantum kernels. Conversely, Pure VQC models are currently ill-suited for high-dimensional tabular data without aggressive (and potentially lossy) dimensionality reduction.

4.2 Discussion

The poor performance of the pure quantum model (51.61%) highlights a critical challenge: encoding high-dimensional sports data (dozens of stats per team) into a small number of qubits (7 in our simulation) results in massive information loss. While PCA helped technically, it likely discarded variance critical for predicting game outcomes.

The Hybrid model's ability to achieve the highest internal accuracy (82.61%) is a significant finding. It suggests that if the "generalization gap" can be closed (perhaps through better regularization or larger training sets) hybrid models could eventually outperform classical ones. The quantum layer may be capturing correlations that the classical layers miss, provided the input features are condensed effectively.

From an ethical standpoint, these models should be used cautiously. While 64% accuracy is profitable for casual analysis, it is insufficient for high-stakes decision-making or gambling, where "black box" neural network predictions can lead to financial harm. Furthermore, the sustainability cost of training quantum models is non-negligible; training the VQC took approximately 4-5x longer than the classical model, consuming more energy for lower accuracy.

One final point that is important to mention is the fact that we used fixed seeds for the different models. Since it’s not really stochastic anymore and we could reproduce the experiments it may mean that we have been lucky while finding our different models.

4.3 Future Work

Future iterations of this project should focus on:

1. Testing the pipelines on actual quantum processors (QPUs) via cloud services (like IBM Quantum) to evaluate the impact of real quantum noise versus simulated environments.
2. Experimenting with amplitude embedding to encode 2^N features into N qubits, allowing the utilization of the full dataset without aggressive PCA.
3. Experimenting with different ansatz.
4. Integrating time-series data (e.g., player fatigue over the season) and weather conditions rather than just season averages to improve external prediction accuracy.
5. Forget about the seeds to make it more stochastic

References

[1] Horvat, T. and Job, J. (2020) The use of machine learning in sport outcome prediction: A review. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery. Available at: https://wires.onlinelibrary.wiley.com/doi/10.1002/widm.1380 [Accessed 15 Dec. 2025].

[2] Bosch, P. (2018) Predicting the winner of NFL-games using Machine and Deep Learning. Thesis, Vrije Universiteit Amsterdam. Available at: https://beta.vu.nl/nl/Images/werkstuk-bosch_tcm235-888637.pdf [Accessed 15 Dec. 2025].

[3] Jolliffe, I. T. and Cadima, J. (2016) Principal component analysis: a review and recent developments. Philosophical Transactions of the Royal Society A, Available at: https://royalsocietypublishing.org/doi/10.1098/rsta.2015.0202 [Accessed 15 Dec. 2025].

[4] Hamadani, B. (2006) Predicting the outcome of NFL games using machine learning. CS229 Project Report, Stanford University. Available at: http://cs229.stanford.edu/proj2006/BabakHamadani-PredictingNFLGames.pdf [Accessed 15 Dec. 2025].

[5] Schuld, M. and Killoran, N. (2019) Quantum Machine Learning in Feature Hilbert Spaces. Physical Review Letters. Available at: https://link.aps.org/doi/10.1103/PhysRevLett.122.040504 [Accessed 15 Dec. 2025].

[6] Farhi, E. and Neven, H. (2018) Classification with Quantum Neural Networks on Near Term Processors. Available at: https://arxiv.org/abs/1802.06002 [Accessed 15 Dec. 2025].

[7] Arthur, D. and Date, P. (2022) A Hybrid Quantum-Classical Neural Network Architecture for Binary Classification. Available at: https://arxiv.org/abs/2201.01820 [Accessed 15 Dec. 2025].

[8] Bunker, R. P. and Thabtah, F. (2019) A machine learning framework for sport result prediction. Applied Computing and Informatics. Available at: https://www.sciencedirect.com/science/article/pii/S2210832717301485 [Accessed 15 Dec. 2025].