0. Significance

By boosting classical ML with qc, we can take advantage of benefits such as better handling of high-dimensional data (this dataset had around 200 columns), more efficient optimization, and more intuitive, simpler models.

I. Data Pre-processing

We wanted to ensure that the immunotoxic data was correct. As such, we wanted to cross-check the 0/1 with a more reliable database. We explored the possibility of using Protox. By providing a SMILES ID of a molecule, the platform informed us whether the molecule was immunotoxic or not (classified as active or inactive), and also provided us with probabilities. For compounds which were tested for immunotoxicity, this probability was ~99%. For new compounds, these greatly varied. Please check the appendix (Figure 1) for further details.

II. Model building

We tried many things in order to get high accuracy, while also watching our confusion matrix to ensure that results were not too skewed. We thought that false positives would be less harmful than false negatives, so we kept that in mind while trying different models.

For one of our simpler exploratory models with 3 qubits each representing 1 of the 3 PCs, the pred. accuracy was 60.42%, with the confusion matrix:

	Pred 0	Pred 1
True 0	50	22
True 1	16	8

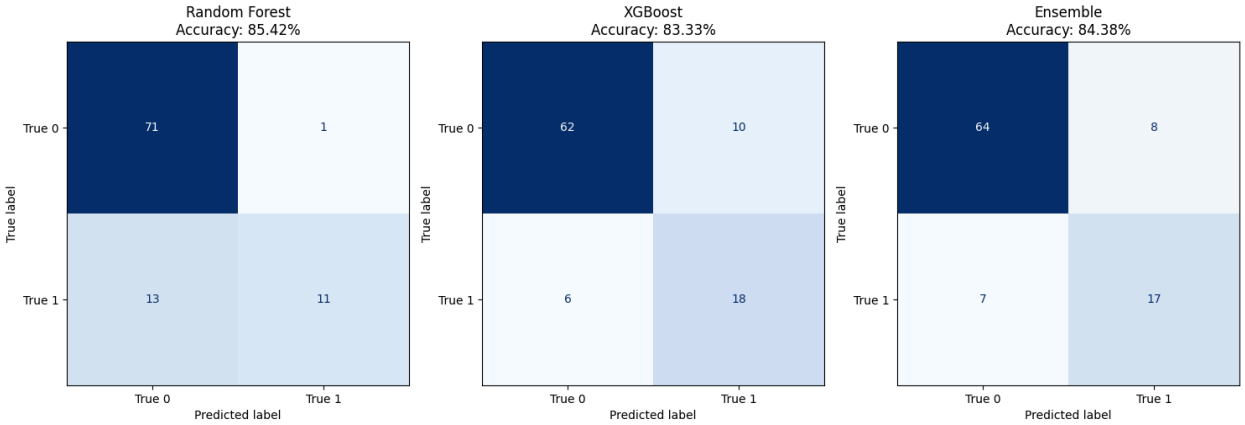
Our classical ML models needed to be highly complicated in order to increase AUC to anywhere significantly over 60%, making them less generalizable and at risk of overfitting.

We knew we could do better.

Given the long run-time for 4 qubits to represent the four top principal components (as these are, after all, classical simulations of quantum systems), we decided to stick to 3 qubits and improve with classical ML techniques.

If you wish to follow along with the steps below, please view our Github. The data pre-processing and exploration is in part_1, while results and analysis are in part_2.

III. Results



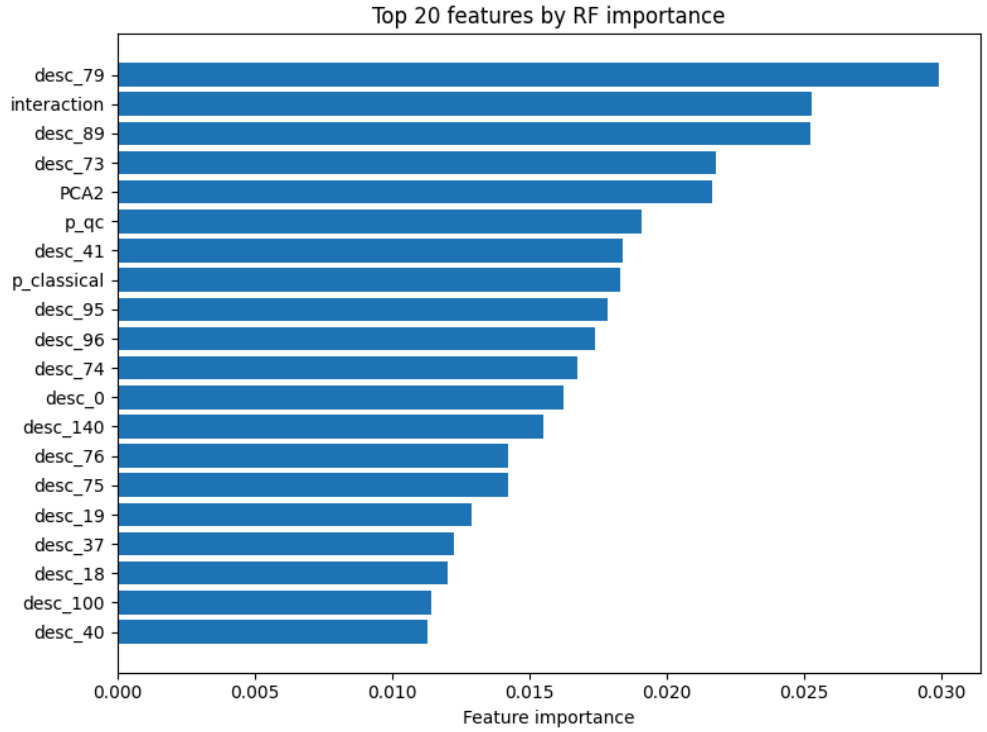
Seeing how all these methods had similar accuracies (and AUC, shown in the Appendix as Figure 2), we decided to proceed with the ensemble method that weighs both Random Forest and XGBoost.

Ensemble Model Quantitative Results

- False Positive Rate: 0.111111
- False Negative Rate 0.291667
- Precision 0.680000
- Recall 0.708333
- F1 Score 0.693878
- Accuracy 0.843750

We wanted to ensure that we **minimized the false negative rate as low as we could**, as we believe that classifying an immunogenic compound as non-immunogenic is more dangerous than the reverse. However, further than 0.29 simply was not possible without causing the false positive rate to increase incredibly high. One of our other competing models had a false negative rate of ~0.10, but a false positive rate around 0.50, which isn't too helpful or insightful. This is the main trade-off that we had to consider.

The top features are listed below.



The different feature names are either the column names (e.g. the 79th column excluding 'Labels' and 'Smiles' would map to desc_79).

For the reader's convenience, desc_79, desc_89 and desc_73 are MOE MR VSA Descriptors, which reveal information about how much of the molecule's surface is available for van der Waals interactions. We theorise, using our biochemical knowledge, that a greater space availability for van der Waals interactions is also correlated with a greater chance of a reaction occuring,[1] [2] which **would explain why they are top features**. PCA2 is just the second principle component piece, and p_qc is the component we derived using the quantum circuit. You can see it is more impactful than p_classical.

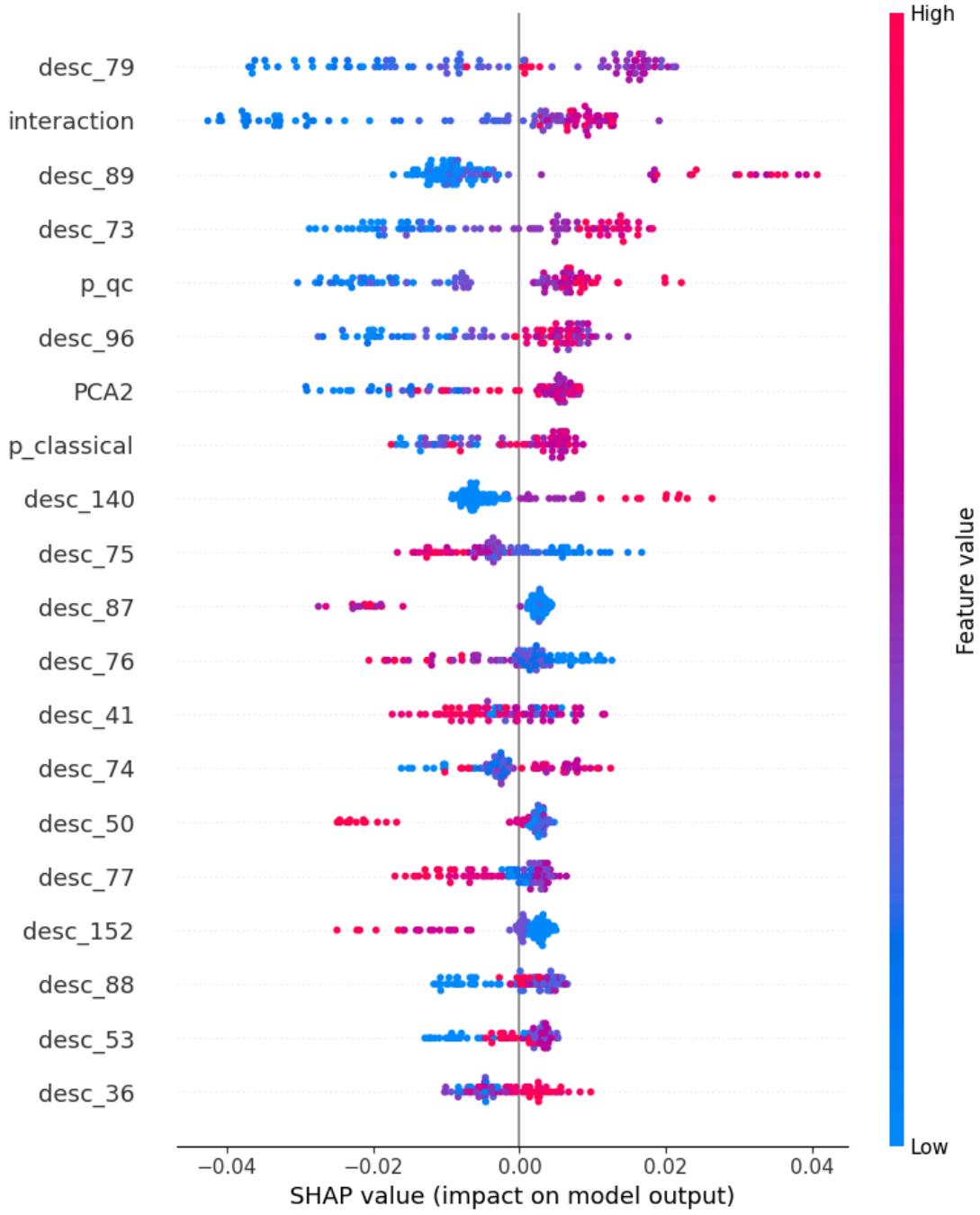
Please note that as none of this molecular data is linked to any sort of confidential information, there is no need for encryption or special steps for the protection of the data. In the case where the specific dataframe regarding the molecular quantities would be needed, we have ensured that the user can locally select a file and run the code on their own device.

---

[1] Tantardini, C., Michalchuk, A.A.L., Samtsevich, A. *et al.* The Volumetric Source Function: Looking Inside van der Waals Interactions. *Sci Rep* **10**, 7816 (2020). https://doi.org/10.1038/s41598-020-64261-4

[2] Kleshchonok, A., Tkatchenko, A. Tailoring van der Waals dispersion interactions with external electric charges. *Nat Commun* **9**, 3017 (2018). https://doi.org/10.1038/s41467-018-05407-x

We also performed a SHAP analysis to get a better understanding of the features.
It is evident that some features, like desc_79, have a wider impact, while others are more 'clustered', like desc_50. It can have a high impact, or a low impact. However most notably, the quantum computing principal component has a higher value for positive (immunoactive) compounds. Please view it on the following page.



IV. Implications

We have successfully built a three-qubit quantum-classical hybrid model that has an accuracy of 84.38% and AUC of 0.880. By using quantum computing, we were able to achieve a more balanced false positive and false negative rate than the solely-classical equivalents (Figure 3 in the Appendix).

In the future, we would like to use quantum computing for this class of problem not only to improve our machine learning models, but in order to solve the more natural problem of stimulating the molecules and determining their immunogenicity by stimulating the chemical interactions they would have with active sites and other biological features.

V. APPENDIX

	SMILES	Target	Shorthand	Prediction	Probability	Label	BalabanJ	BertzCT	Chi0	Chi0n	...	fr_sulfide	fr_sulfonamd	fr_sulfone	fr_term_a
0	COC(=O)N(C)c1c(N)nc(nc1N)c2nn(Cc3ccccc3F)c4ncc...	Immunotoxicity	immuno	Active	0.80	0	1.821	1266.407	22.121	16.781	...	0	0	0	
1	C[C@H](N(O)C(=O)N)c1cc2ccccc2s1	Immunotoxicity	immuno	Inactive	0.93	0	2.363	490.434	11.707	8.752	...	0	0	0	
2	C[N+](C)(C)CC(=O)[O-]	Immunotoxicity	immuno	Inactive	0.99	0	3.551	93.092	6.784	5.471	...	0	0	0	
3	CC(C)n1c(\C=C\[C@H](O)C[C@H](O)CC(=O)O)c(c2ccc...	Immunotoxicity	immuno	Inactive	0.86	1	2.076	1053.003	21.836	16.995	...	0	0	0	
4	C\C(=C(\C#N)/C(=O)Nc1ccc(cc1)C(F)(F)F)\O	Immunotoxicity	immuno	Inactive	0.99	1	2.888	549.823	14.629	9.746	...	0	0	0	
5	CC(C)c1nc(CN(C)C(=O)N[C@H](CCN2CCOCC2)C(=O)N[C...	Immunotoxicity	immuno	Inactive	0.95	0	1.277	1701.679	38.125	31.447	...	0	0	0	
6	CCOC(=O)C1=C(C)NC(=C([C@@H]1c2cccc(Cl)c2Cl)C(=...	Immunotoxicity	immuno	Inactive	0.99	0	2.739	783.281	18.723	14.405	...	0	0	0	
7	CC(C)NC[C@H](O)COc1ccc(COCCOC(C)C)cc1	Immunotoxicity	immuno	Inactive	0.94	1	2.197	405.988	17.079	14.749	...	0	0	0	
8	COc1c(N2C[C@@H]3CCCN[C@@H]3C2)c(F)cc4C(=O)C(=C...	Immunotoxicity	immuno	Inactive	0.99	0	1.729	1048.298	20.284	16.281	...	0	0	0	
9	Cc1c(C)c2O[C@](C)(COc3ccc(C[C@@H]4SC(=O)NC4=O)...	Immunotoxicity	immuno	Inactive	0.90	0	1.405	1049.171	22.336	18.204	...	1	0	0	
10	CCN(CC)CC(=O)Nc1c(C)cccc1C	Immunotoxicity	immuno	Inactive	0.98	0	2.680	363.559	12.836	11.209	...	0	0	0	
11	N[C@H](CCCN=C(N)N)C(=O)O	Immunotoxicity	immuno	Inactive	0.99	0	3.440	176.387	9.560	6.733	...	0	0	0	
12	CCCCN1CCCC[C@@H]1C(=O)Nc2c(C)cccc2C	Immunotoxicity	immuno	Inactive	0.99	0	2.065	463.857	15.242	13.615	...	0	0	0	
13	NC(=O)C([C@@H]1CCN(CCc2ccc3OCCc3c2)C1)(c4ccccc...	Immunotoxicity	immuno	Inactive	0.85	0	1.382	1041.601	21.968	18.374	...	0	0	0	
14	NCCCC[C@H](N[C@@H](CCc1ccccc1)C(=O)O)C(=O)N2CC...	Immunotoxicity	immuno	Inactive	0.99	1	2.079	682.154	21.225	16.626	...	0	0	0	
15	CC(=O)S[C@@H]1CC2=CC(=O)CC[C@]2(C)[C@H]3CC[C@@...	Immunotoxicity	immuno	Active	0.83	1	1.557	817.698	20.604	17.792	...	1	0	0	

Figure 1: Data pre-processing: augmenting the Protox predictions and probabilities to the dataframe. Available in the part_1 notebook.

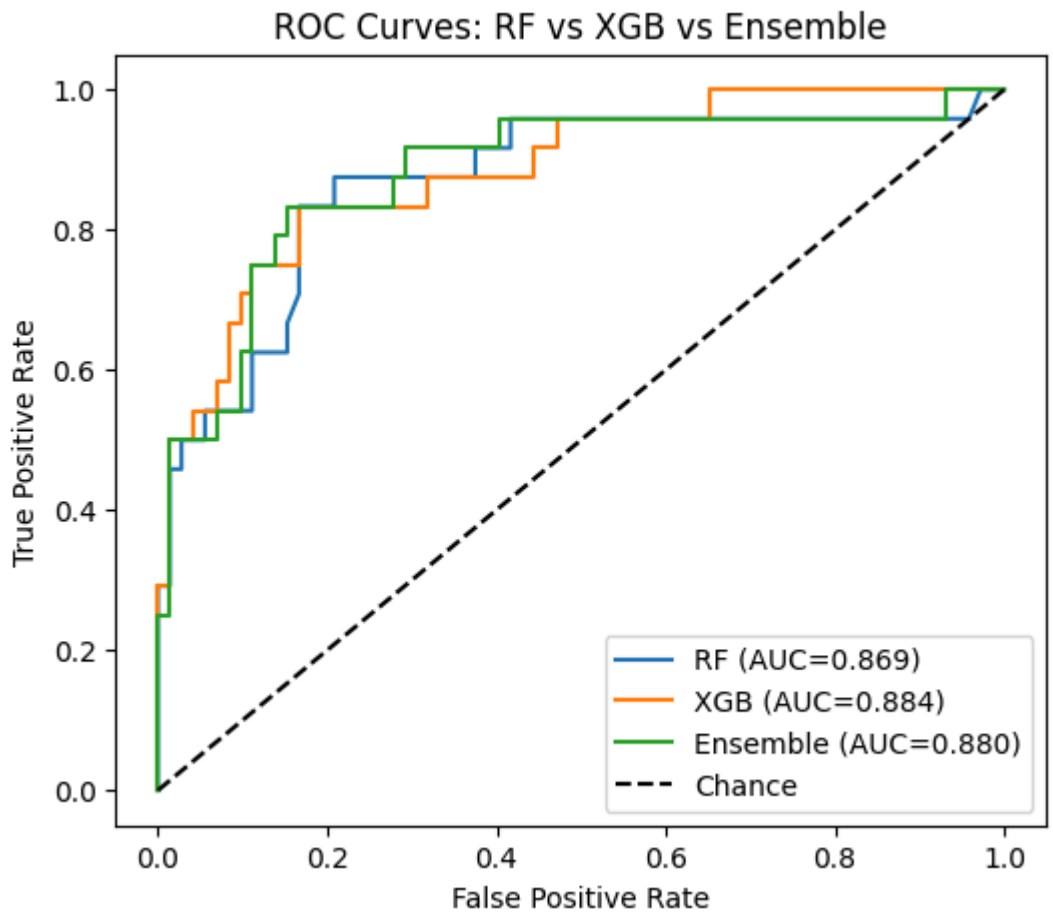


Figure 2: AUC curves of the different methods. Note that they are all comparable. Available in the part_2 notebook.

```
Confusion Matrix (with balanced LR):
        Pred 0  Pred 1
True 0       8      64
True 1       0      24

Metrics:
                          Value
Accuracy                0.333333
Precision               0.272727
Recall (TPR)            1.000000
F1 Score                0.428571
False Positive Rate     0.888889
False Negative Rate     0.000000
```

Figure 3: Confusion Matrix and Stats of the purely Classical Model, available at the end of the part_2 notebook.

If you would like to try running part 2 of the code yourself, feel free to try this colab:
Exploration_of_QML_part_2_results_and_analysis.ipynb