Project 6: Inverse Design of Optimal Airfoil Geometries

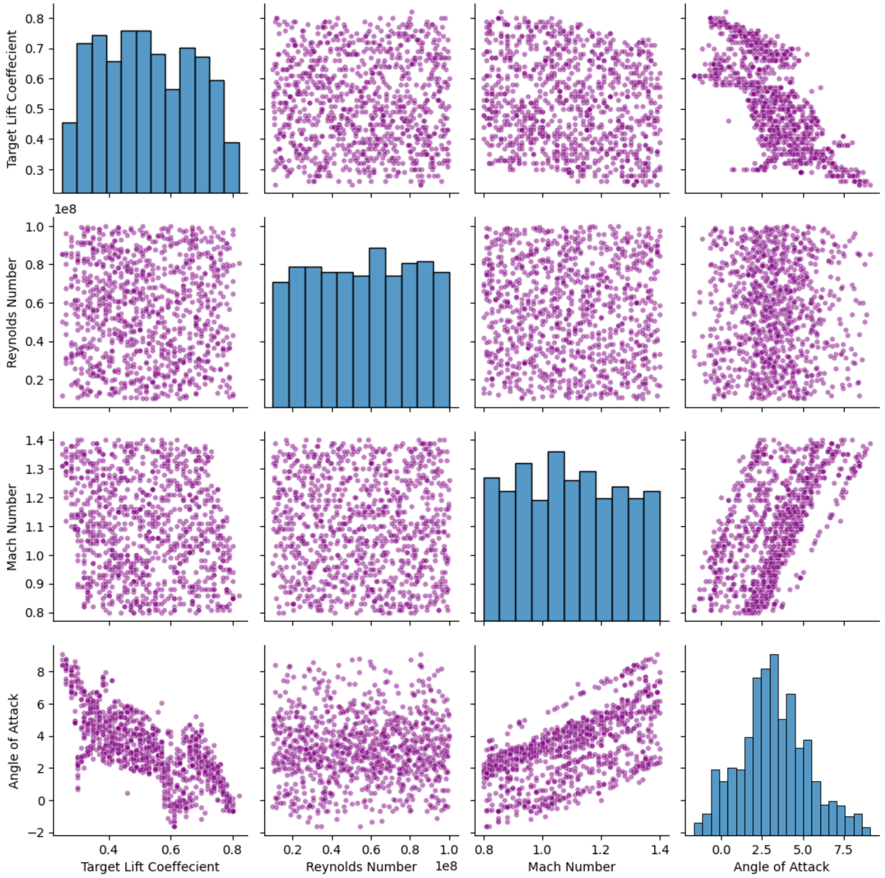
PRESENTERS:
ISHAN DUTTA AND HARVIR GHUMAN

Problem Overview

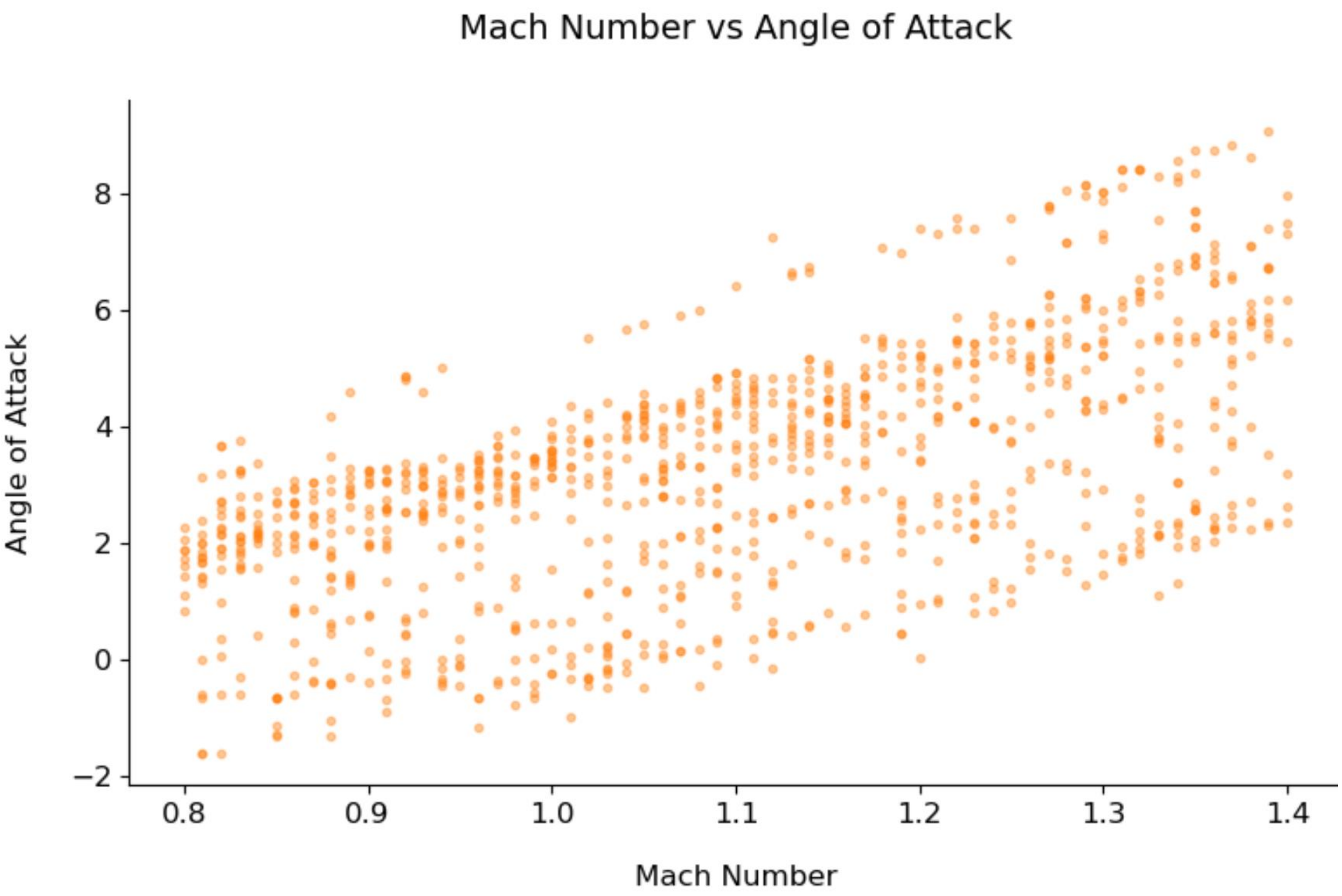
Use an existing data set to train and test a model that can predict airfoil geometries based on the given boundary conditions:

- Angle of attack
- Mach Number
- Reynold's Number
- Target Lift Coefficient

Data Visualization

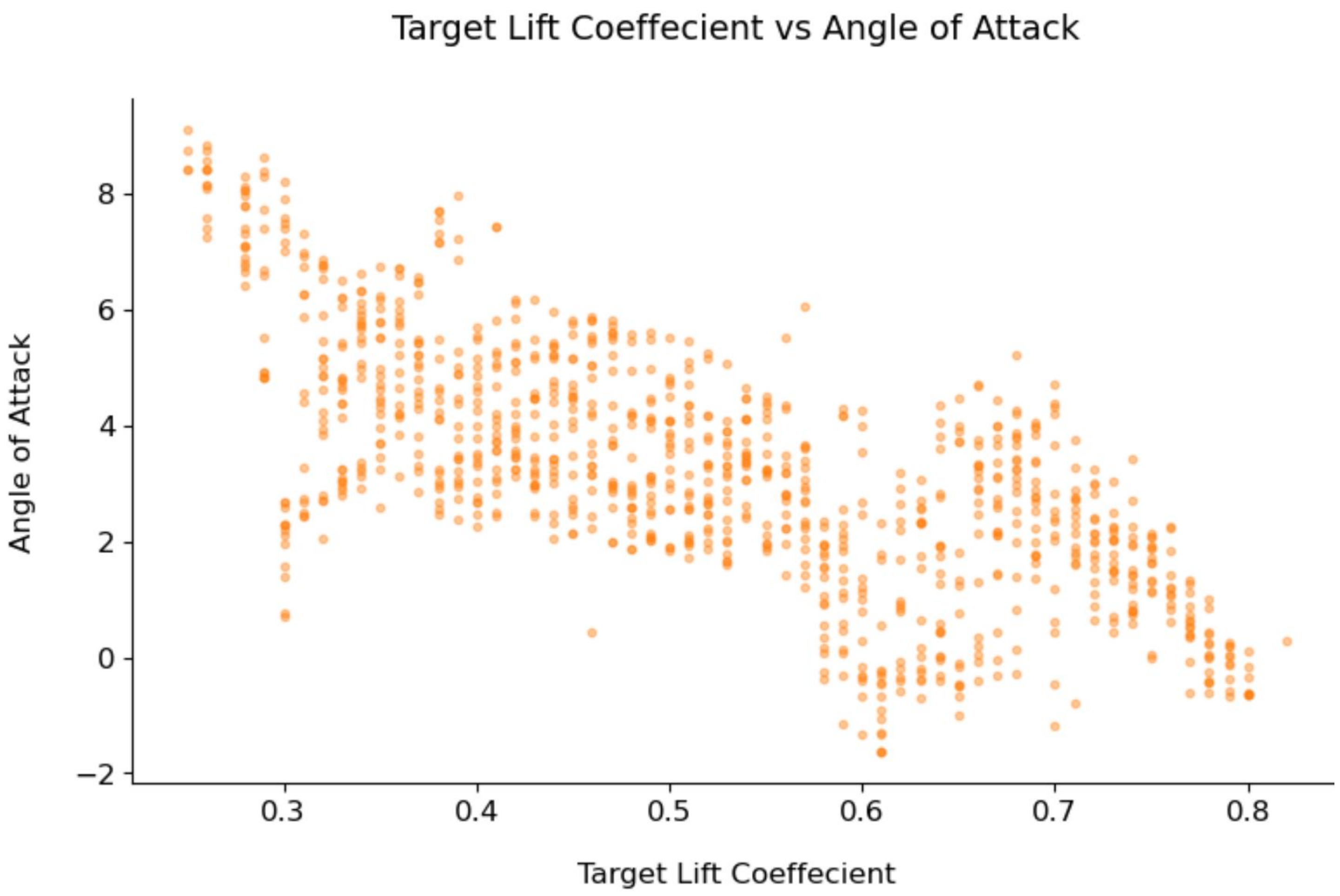


- Feature Relationship showing correlations between boundary conditions



Data Visualization

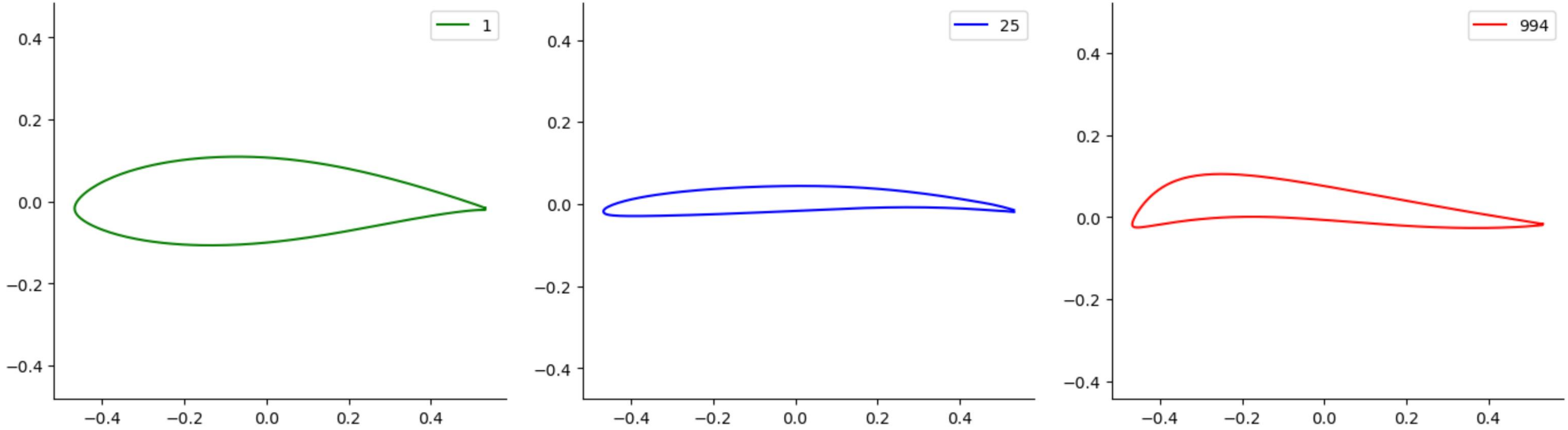
- Slight Linear correlation
- There is no clear physical relationship between Mach number and Angle of attack



Data Visualization

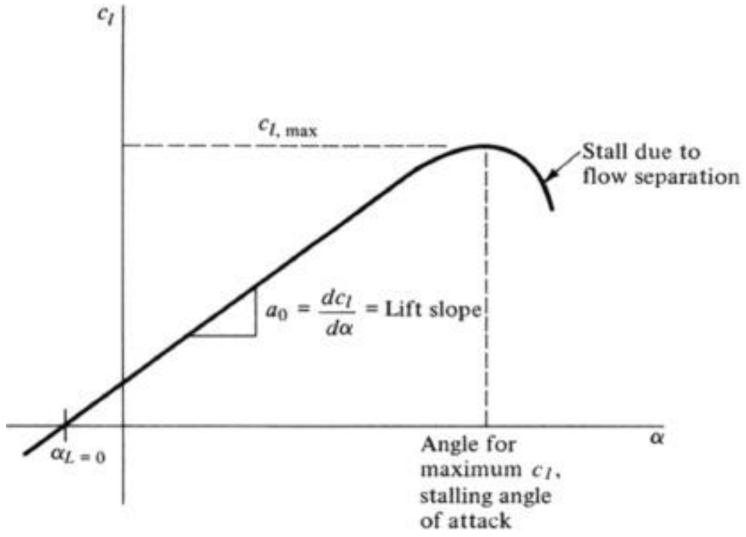
- Slight Linear correlation
- Lift Coefficient and angle of attack have negative correlation, but generally it is positive

Data Visualization



Airfoil geometries vary from simple to extremely complex and impractical

Visualization Insights



- Airfoil Geometry: High Dimensional Representation
  - Each Airfoil represented by 192 (x, y) pairs
  - Dimensionality reduction necessary to reduce model complexity and speed up convergence
  - Wide variety of airfoil geometries (transonic to supersonic regime)

- These linear correlations did not seem to apply accurately to real world
  - Complex, impractical airfoil geometries

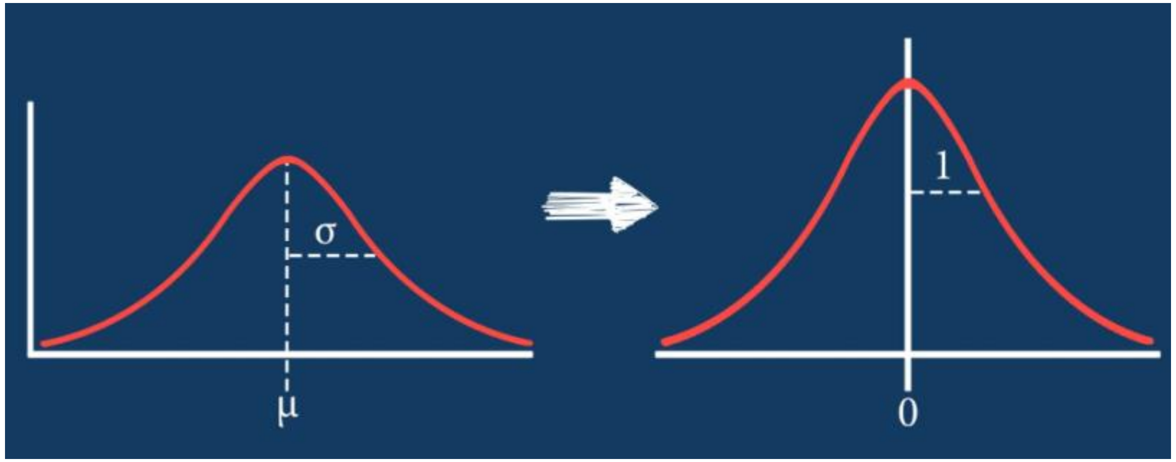
- Generally, 2 boundary conditions won't always have a strong linear correlation even if they are related

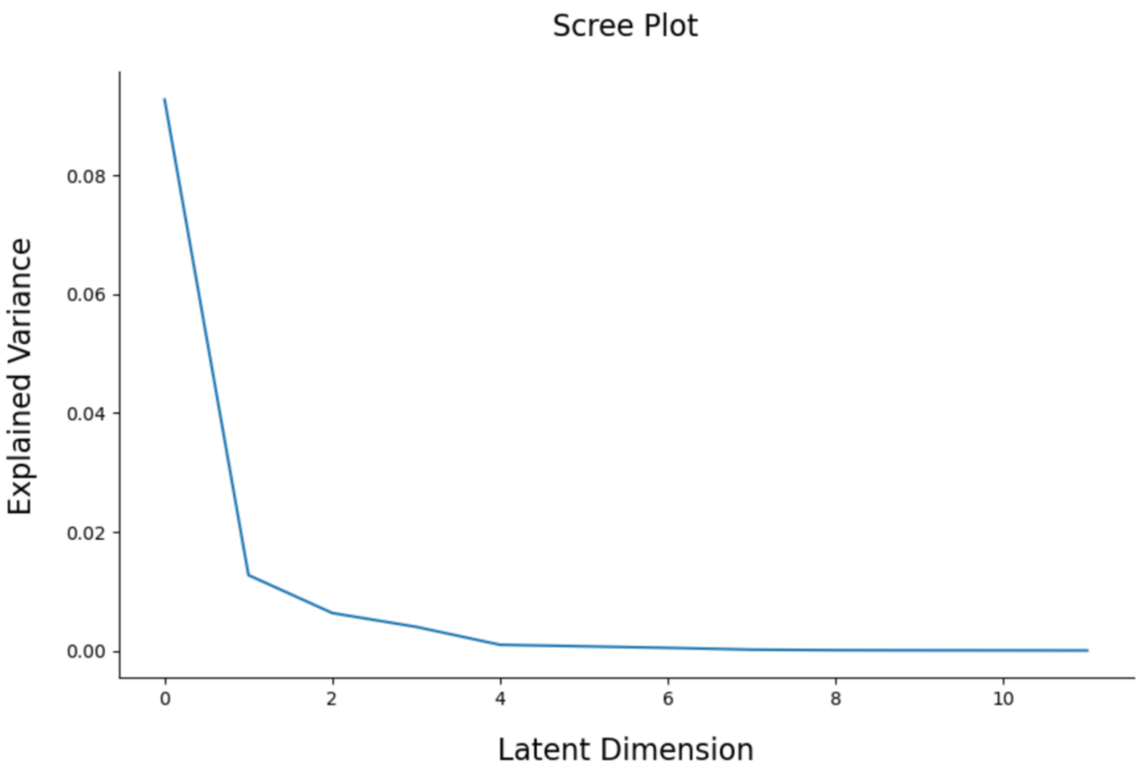
- Hard to visualize relationship between boundary conditions and (x, y) geometry without physics model

Data Preprocessing

- 4 input features: Mach Number, Reynolds number, Lift Coefficient, Angle of Attack
  - Mach Number: 0.8 - 1.4
  - Reynolds Number: 0.2 - 1.0E8
  - Lift Coefficient: 0.3 - 0.8
  - Angle of Attack: -2 – 9 degrees

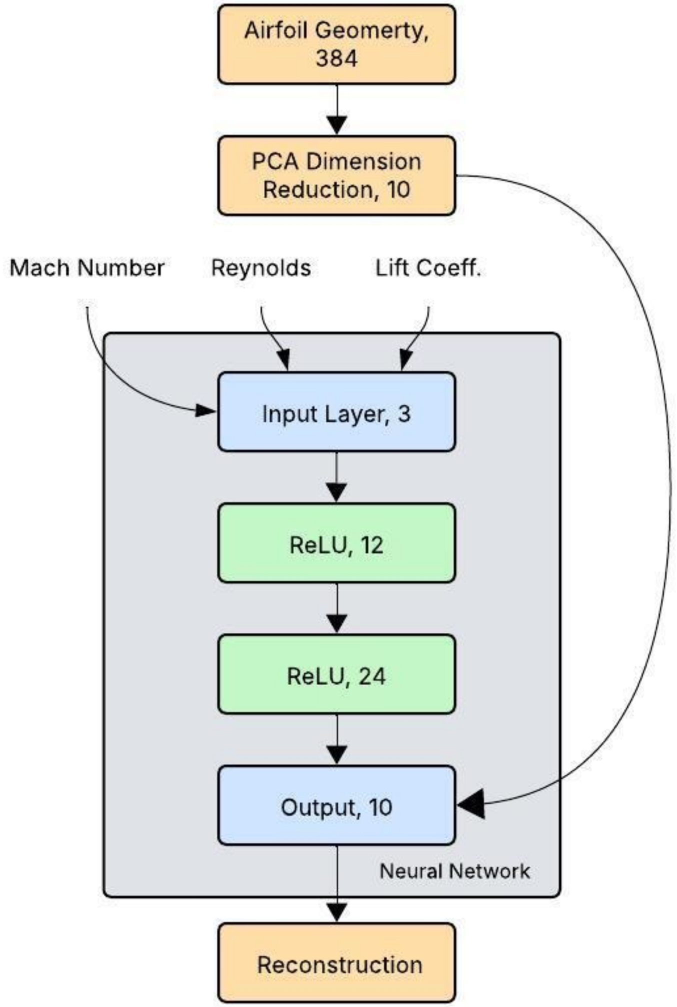
- Standardized above input features to have a mean of 0 and standard deviation of 1
- No feature scaling performed on Airfoil Geometry





Baseline Model

- PCA Dimensionality Reduction:
  - Reduce Airfoil Geometry size:
    - Initial: 995 x 192 x 2 (Airfoil x Point x coordinates)
    - Final: 10
- 80-20 Train Test Split
- No Feature Scaling

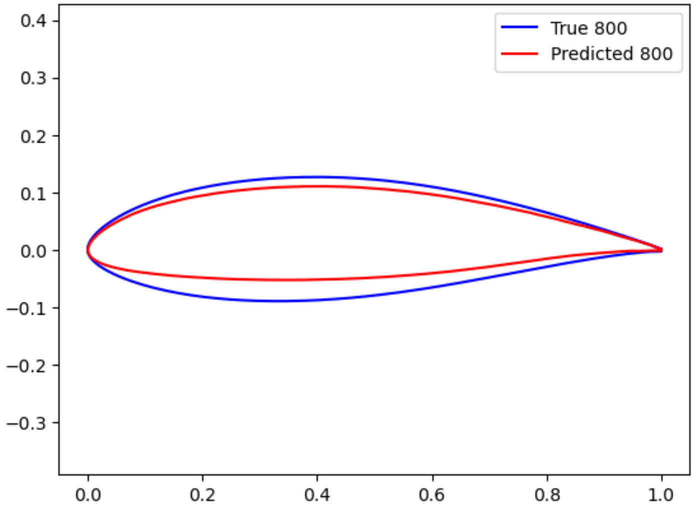


Baseline Model

- Deep Neural Network
  - Input Layer: 3 Neurons
  - Hidden Layer 1 (ReLu): 12 Neurons
  - Hidden Layer 2 (ReLu): 24 Neurons
  - Output Layer: 10 Neurons
- Adam Optimizer:
  - Learning Rate: 0.1
- MSE Loss Function
- Training:
  - N = 100 Epochs
  - Batch size = 16

Baseline Model Results

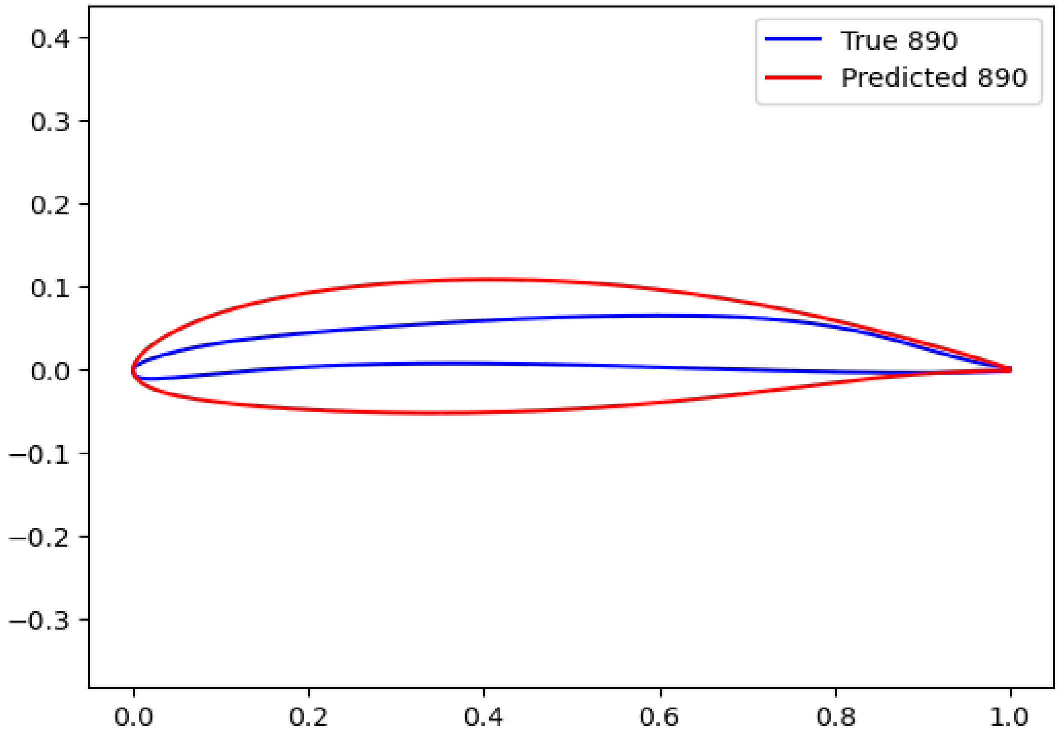
```
Epoch 0: Train Loss = 0.0316, Test Loss = 0.0255
Epoch 20: Train Loss = 0.0020, Test Loss = 0.0015
Epoch 40: Train Loss = 0.0019, Test Loss = 0.0016
Epoch 60: Train Loss = 0.0019, Test Loss = 0.0016
Epoch 80: Train Loss = 0.0019, Test Loss = 0.0016
Epoch 100: Train Loss = 0.0020, Test Loss = 0.0016
```



- High learning rate could have led to overshoot in minimization of cost function
- Model is underfitting due to lack of feature scaling and poor hyperparameter tuning
- Batch size too small leading to noisy gradients

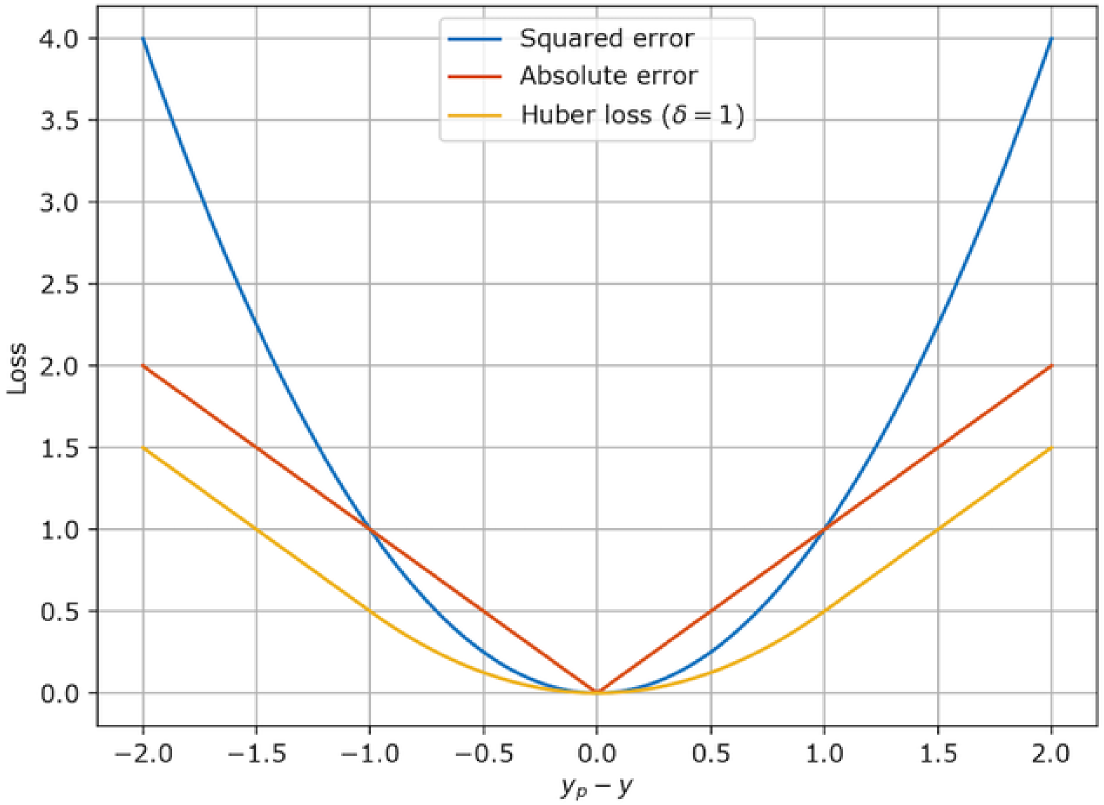
Baseline Model Insights

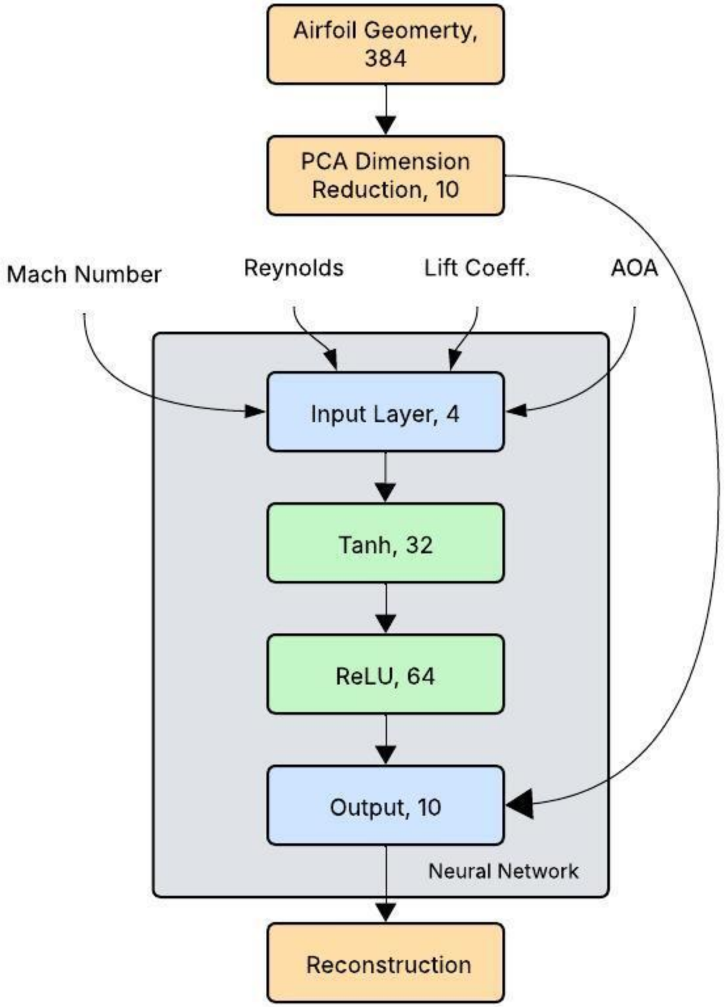
- Regularization could help improve generalization on unseen data
- MSE was penalizing large errors too severely leading to inaccurate generalizations
- Airfoil 890 is an example of the model not generalizing well and memorizing instead
  - Cross validation can help the model account for more complex geometries



Final Model

- PCA Dimensionality Reduction:
  - reduce Airfoil Geometry size from 995 x 384 to latent dimension of 10
- K – Fold CV
  - K = 6 Folds
  - 10 % held out of CV for testing on unseen data
- Huber Loss Function
  - Reduce impact of outliers while sensitive to small errors
  - Combination of MSE and MAE
  - Delta = 0.001

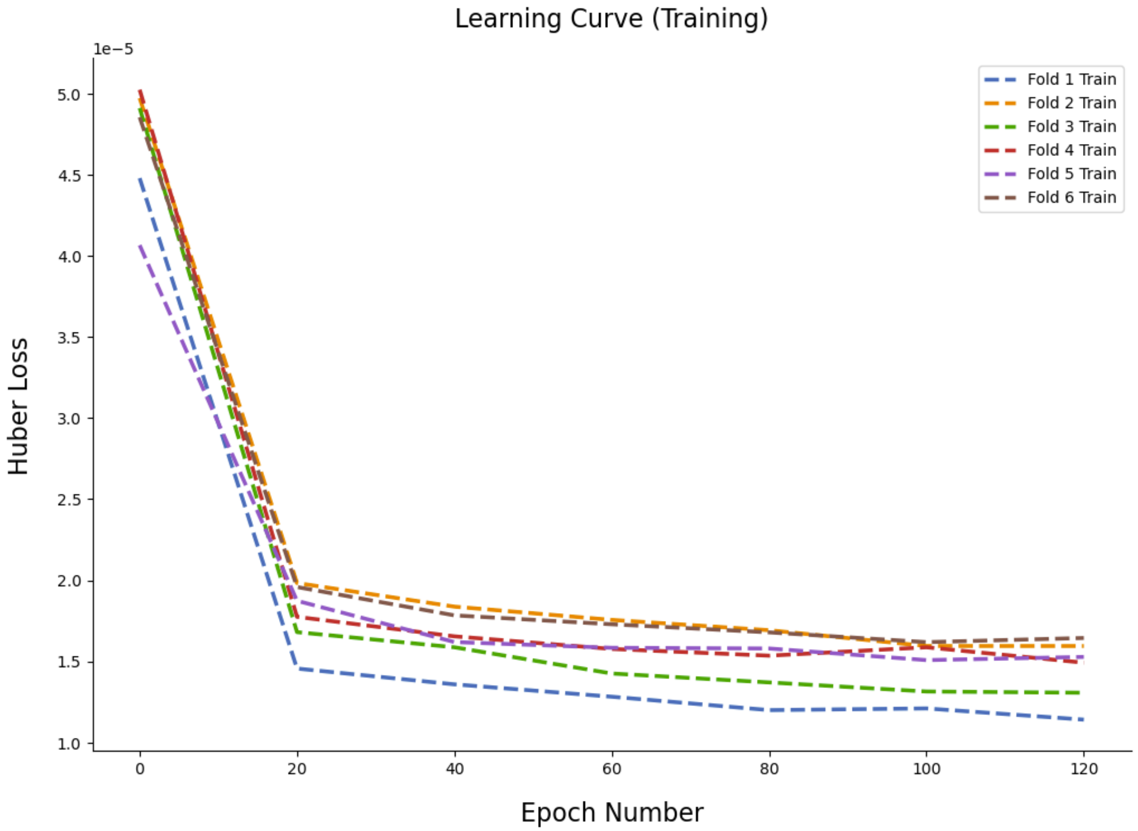




Final Model

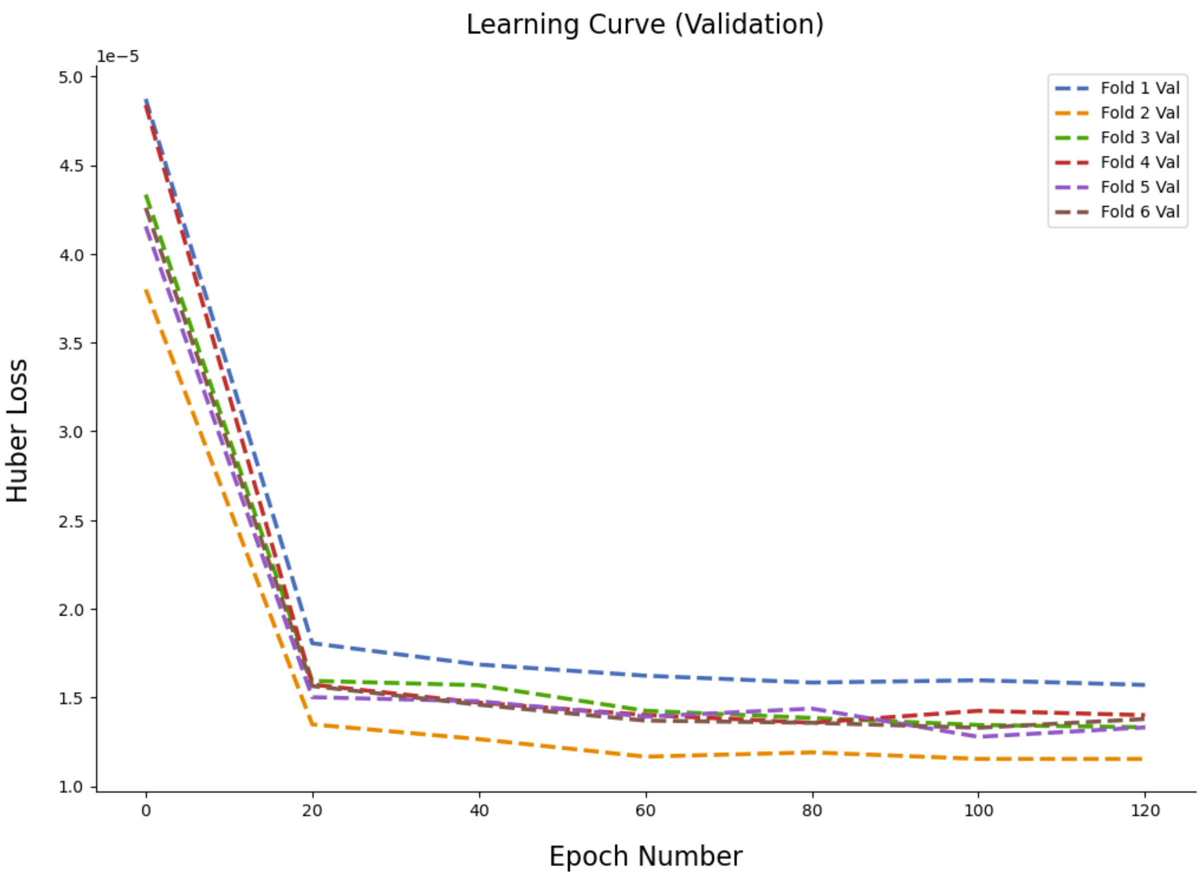
- Deep Neural Network
  - Input Layer: 4 Neurons
  - Hidden Layer 1 (Tanh): 32 Neurons
  - Hidden Layer 2 (ReLu): 64 Neurons
  - Output Layer: 10 Neurons
- Adam Optimizer:
  - Learning Rate: 0.01
  - Weight Decay: 1e-6
    - L2 Regularization but weight update rule
- Training:
  - N = 120 epochs
  - Batch Size = 72

Final Model



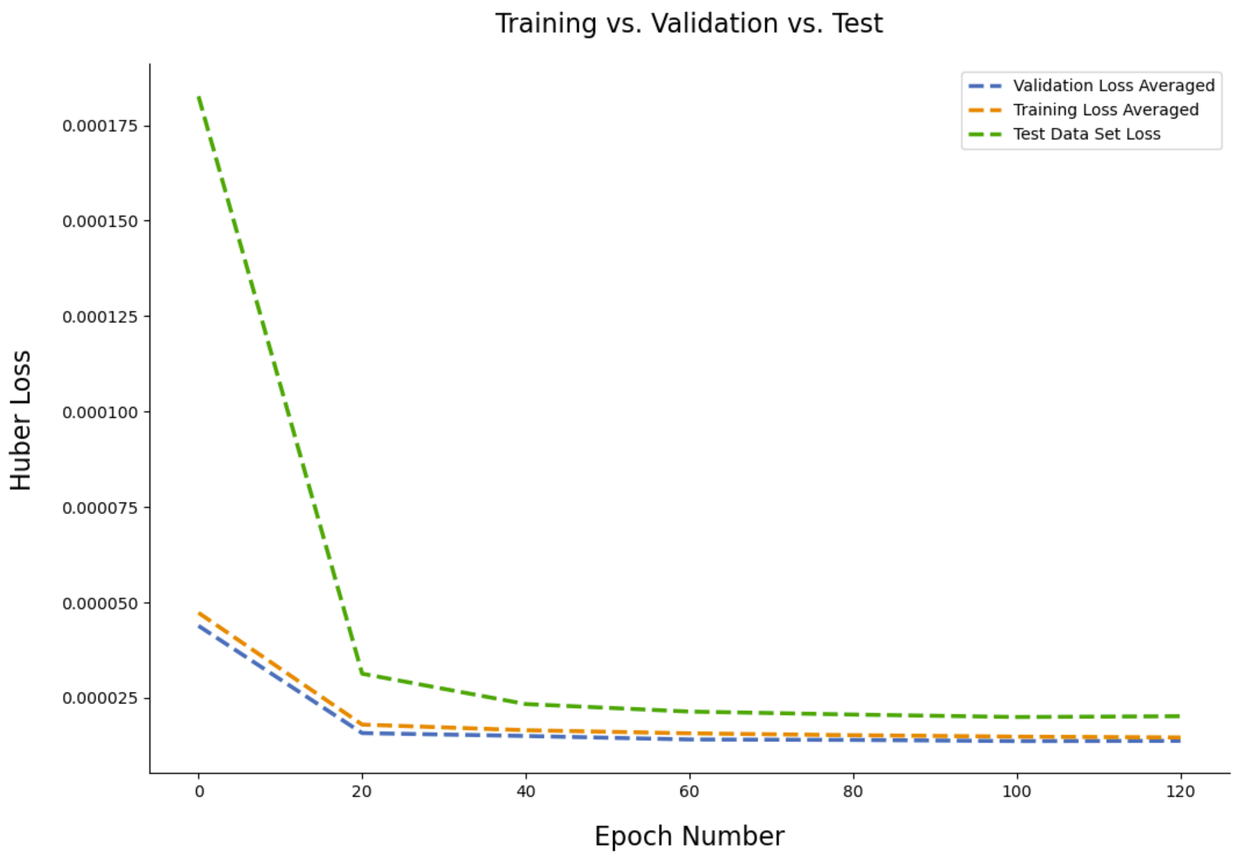
Learning Curve (Training)
1e−5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Huber Loss
0
20
40
60
80
100
120
Epoch Number
Fold 1 Train
Fold 2 Train
Fold 3 Train
Fold 4 Train
Fold 5 Train
Fold 6 Train

Final Model



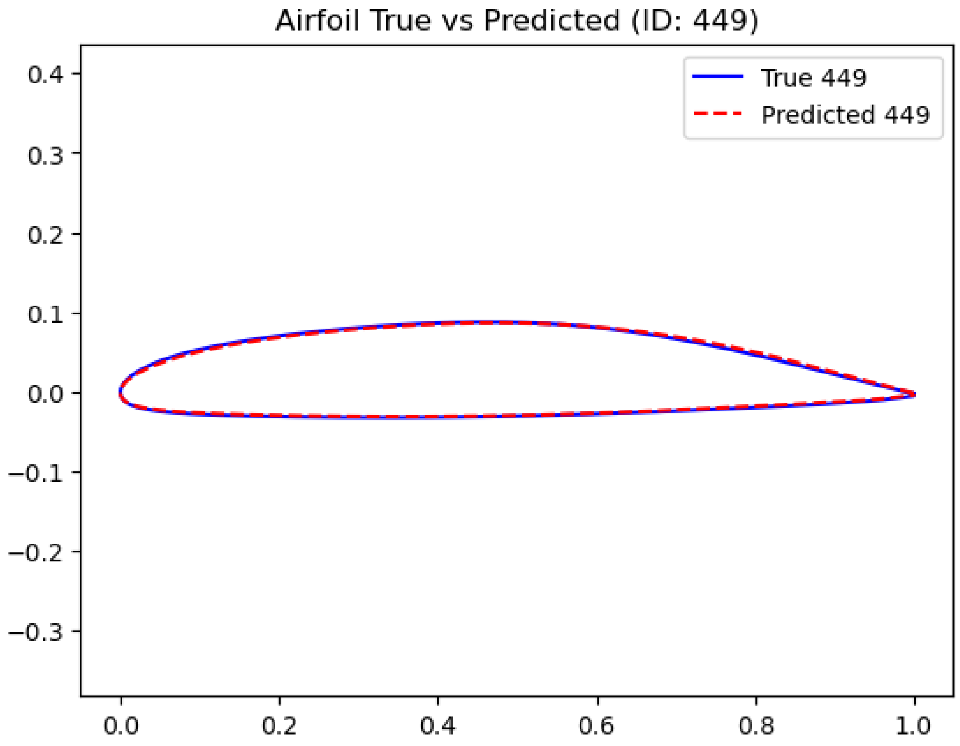
Learning Curve (Validation)
1e−5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
Huber Loss
0
20
40
60
80
100
120
Epoch Number
Fold 1 Val
Fold 2 Val
Fold 3 Val
Fold 4 Val
Fold 5 Val
Fold 6 Val

Final Model

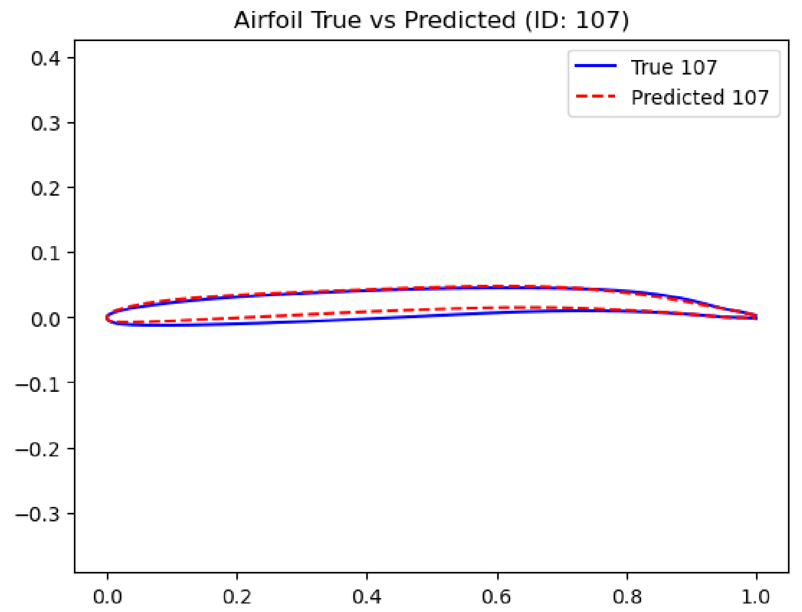

Training vs. Validation vs. Test
Validation Loss Averaged
Training Loss Averaged
Test Data Set Loss
Huber Loss
0.000175
0.000150
0.000125
0.000100
0.000075
0.000050
0.000025
0
20
40
60
80
100
120
Epoch Number

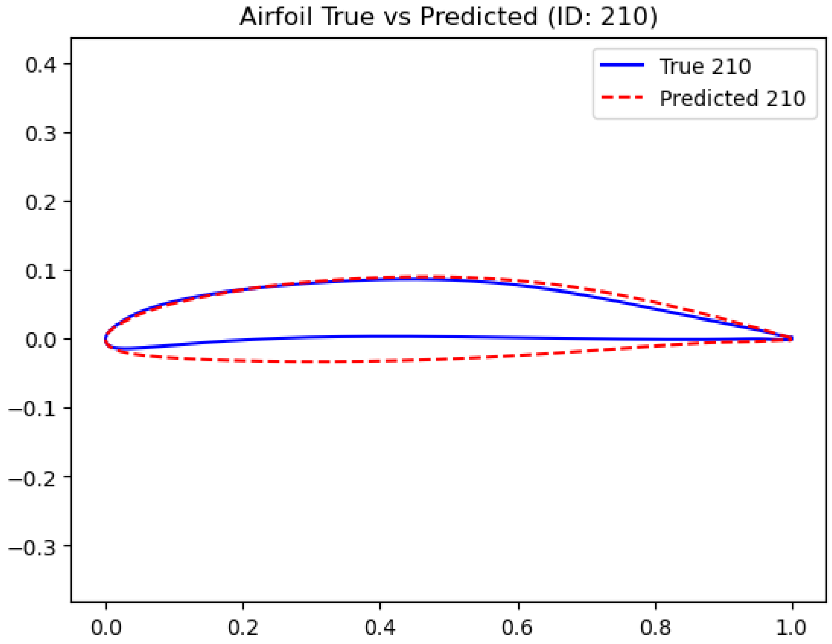
Final Model Insights

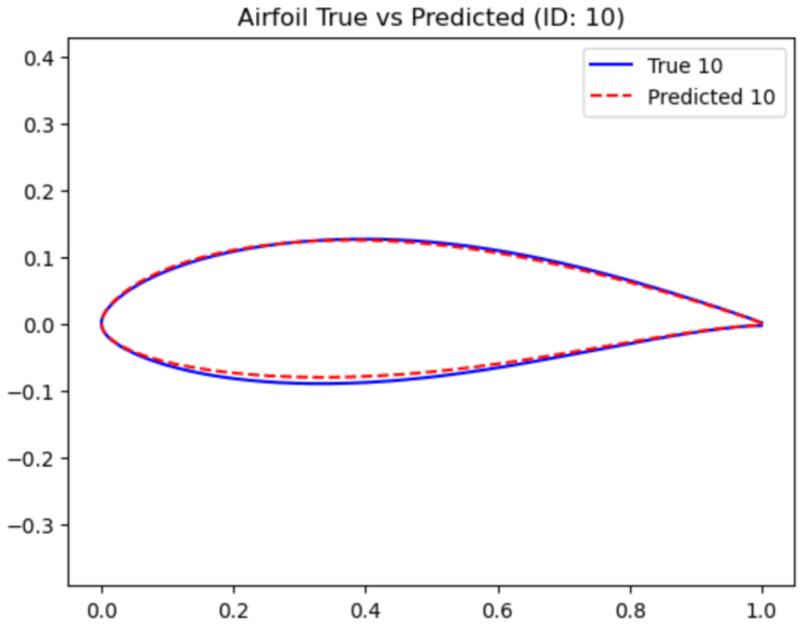
- Weight Decay helped improve generalization
- Learning rate was tuned better to reduce overshoot
- Change from MSE to Huber improved robustness
- This is a dimension reduction coupled with a nonlinear regressor problem
- K Fold CV guaranteed that model saw and learned more complex geometries it hadn't before
- Still not perfect – overshoot of shape
  - Limitations of PCA



Final Model Insights



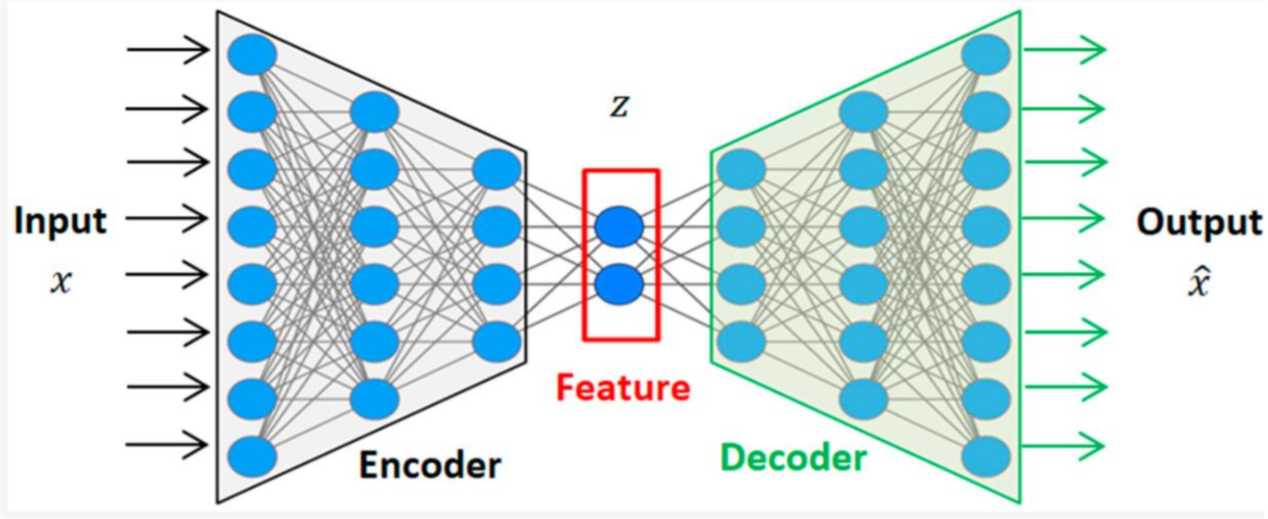




Reconstruction Error MSE
(Original Space): 0.000073

Future Work

- Model could have better data compression to capture nonlinearity of geometry in reduced space
- Autencoder could help with the nonlinearity of data compression and reconstruction
- Bigger dataset or wider variety of airfoils to evaluate true performance of model and possibly more design parameters
- Computer Vision Model



Questions?

Contributions

- Thanks to Harvir Ghuman for:
  - Building the PCA pipeline code for dimensionality reduction
  - Building the final neural network model architecture and tuning hyperparameters
  - Creating insightful data visualizations of feature relationships
  - Testing final model on test data set
  - Creating data visualizations showcasing our final model performance

- Thanks to Ishan Dutta for:
  - Building our baseline neural network architecture in PyTorch
  - Testing our baseline model on test data
  - Providing his expertise in Aerospace engineering to our data visualization results
  - Implementing a new cross validation strategy in the final model
  - Communicating our baseline model results to audience

Github Link

https://github.com/Hghn02/Inverse-Design-of-Optimal-Airfoil-Geometry