STAT 451 PROJECT

# Prediction of Failure in Steel Frames


**Group 6**

Hyeyoung Koh, Civil and Environmental Engineering

Erika Yeonseo Park, Statistics

Jina Yang, Statistics


December 5, 2020

## 1. INTRODUCTION

# Steel Structures and Components

Olivier-Charbonneau Bridge in Quebec

https://www.canambridges.com/projects/highway-25-bridge/

Gable-frame metal building

https://renegadesteelbuildings.com/

Steel frame

Building connections

(a) Tee connection (b) Shear end plate

Photos are adopted from "Unified Design of Steel Structures" by Geschwindner et al., 2017

# 1. INTRODUCTION
# Failure of Steel Structures

**Prevention of failure** is the goal of structural design.

**Structural failures** usually occur by

- loading (gravity, earthquake, wind, fire, etc)
- **Uncertainties** in construction and fabrication
- E.g., material/geometric properties

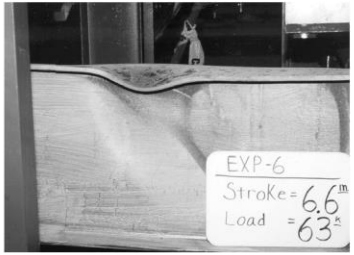
## Types of structural failure

Beam mechanism

Sway mechanism

Photos are adopted from "Unified Design of Steel Structures" by Geschwindner et al., 2017

**Beam**




**Column**

**Connection**

https://s3da-design.com/possible-types-of-failures-in-a-steel-structure/

## 1. INTRODUCTION

# Motivation and Objective

Prevention of failure is the goal of structural design.

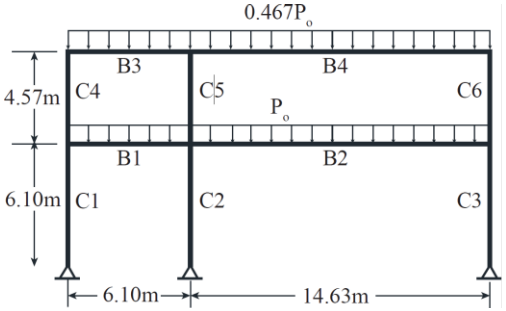
### MOTIVATION

- AISC 360 does not consider the uncertainty.
- Uncertainty affects system performance.
- The level of safety

### OBJECTIVE

- Develop algorithms that predict whether the steel frames experience the failure or not
- Consideration of the uncertainties

### EXAMPLE FRAMES




| | |
|---|---|
| Frame 1 | buckling of a column |
| Frame 2 | progressive yielding |

# 2. EXPERIMENT

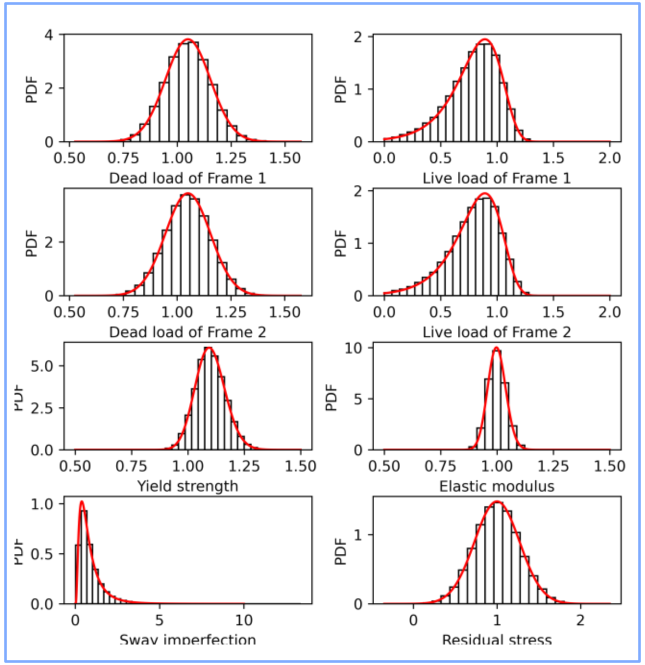
# Dataset

## Distributions of random features (uncertainties)

| Variable | Mean | COV | Distribution | References |
|---|---|---|---|---|
| Dead load ($D$) | $1.05D_n$ | 0.1 | Normal | Ellingwood et al. 1982 |
| Live load ($L$) | $L_n$ | 0.25 | Extreme Value | Ellingwood et al. 1982 |
| Yield strength ($F_y$) | $1.1F_{yn}$ | 0.06 | Lognormal | Bartlett et al. 2003 |
| Elastic modulus ($E$) | $E_n$ | 0.04 | Lognormal | Bartlett et al. 2003 |
| Sway imperfection (ψ) | 1/700 | 0.875 | Lognormal | Lindner and Gietzelt 1984 |
| Residual stress ($X$) | 1.064 | 0.27 | Normal | Shayan et al. 2014 |

Nominal values: $\mathrm{F_{yn}} = 248\ \mathrm{MPa},\ \mathrm{E_n} = 200\ \mathrm{GPa},\ \mathrm{D_{n1}} = 31.1\ \mathrm{kN/m},\ \mathrm{D_{n2}} = 30.4\mathrm{kN/m},$
$\mathrm{L_{n1}} = 46.6\ \mathrm{kN/m},\ \mathrm{L_{n2}} = 45.6\ \mathrm{kN/m}$

- Monte Carlo sampling method was utilized.
- 50,000 samples are generated.



## 2. EXPERIMENT

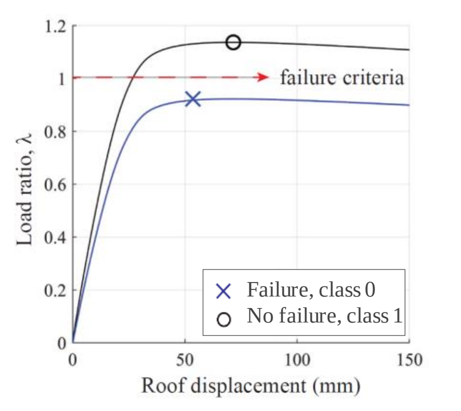
# Dataset

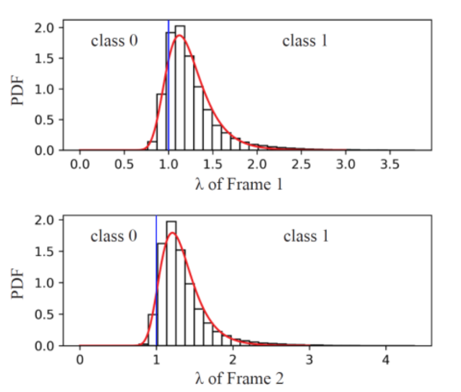
### How to obtain the dataset

- 50,000 finite element analyses were conducted in OpenSees with CHTC
- Obtained load-displacement curves
- Examples that caused convergence errors were removed
- Normalized by dividing nominal values
- 6 random features with 2 class labels

### Number of examples

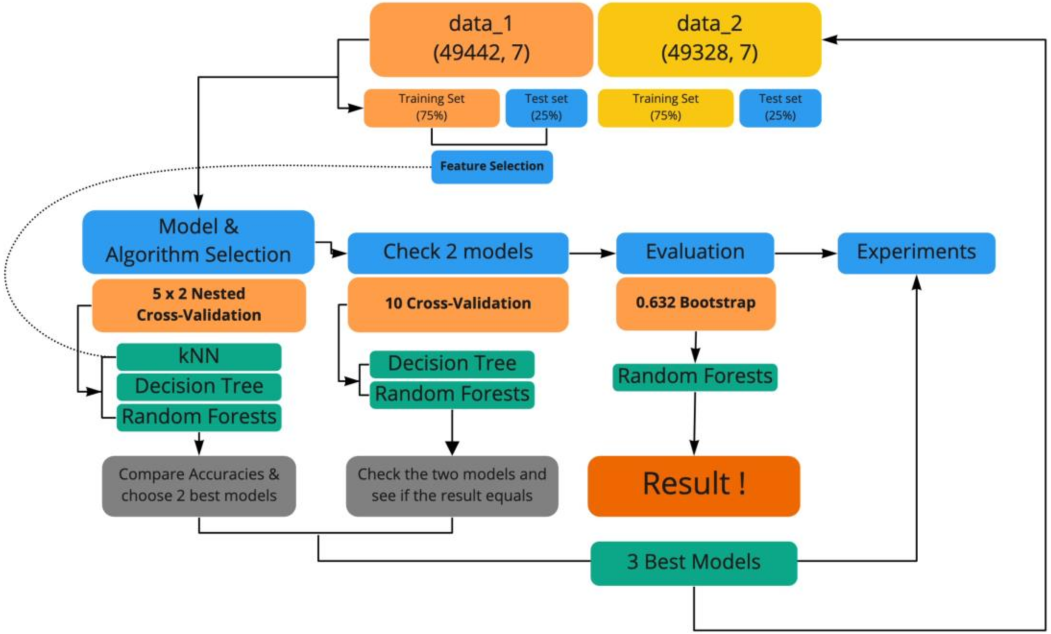
| Failure | Class | Frame 1 | Frame 2 |
|---|---|---|---|
| Failure | Class 0 | 2424 | 7351 |
| No failure | Class 1 | 47019 | 41978 |
| Total | | 49,443 | 49,329 |



## 2. EXPERIMENT

# General Procedure




1. Use data_1
2. Feature selection for *k*NN
3. Model & algorithm selection

   (5 x 2 Nested CV)
4. Check 2 models (2 highest Acc)

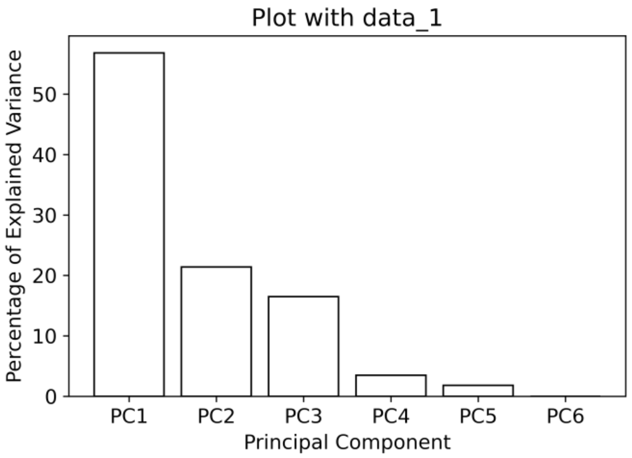
   (10 – CV)
5. Evaluation on the chosen model
6. Use the three models on data_2

## 2. EXPERIMENT

# Feature Selection (PCA + SFS)

### Principal Component Analysis (PCA)

- Explained Variance




### Sequential Feature Selector (SFS)

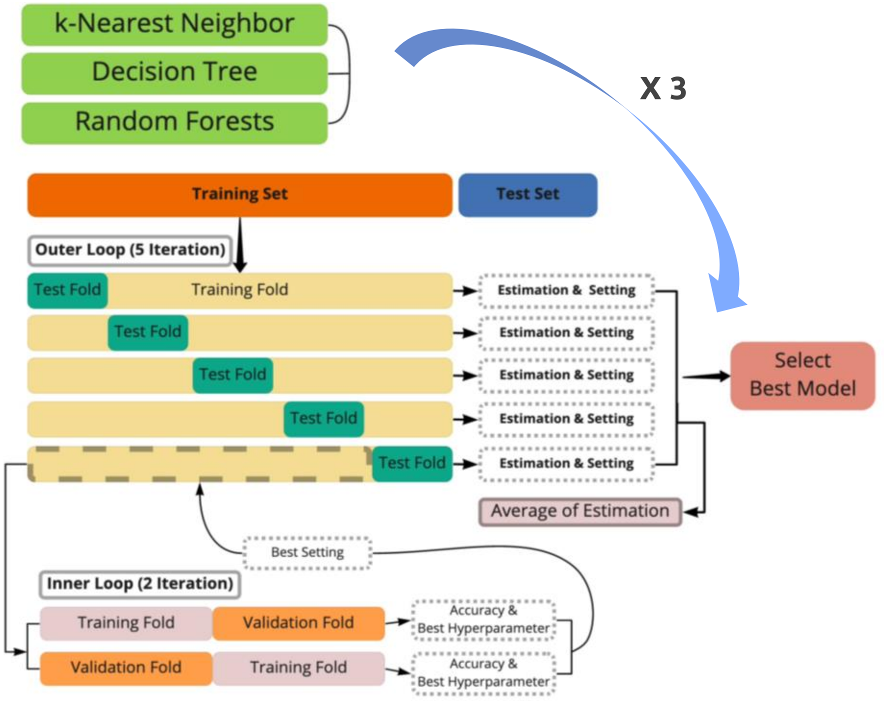
```
{1: {'feature_idx': (1,),
  'cv_scores': array([0.92237369]),
  'avg_score': 0.9223736903846932,
  'feature_names': ('1',)},
 2: {'feature_idx': (1, 5),
  'cv_scores': array([0.94090045]),
  'avg_score': 0.9409004490109624,
  'feature_names': ('1', '5')},
 3: {'feature_idx': (0, 1, 5),
  'cv_scores': array([0.95251001]),
  'avg_score': 0.952510011730917,
  'feature_names': ('0', '1', '5')}}
```

- **Selected features:**

  ['Dead load', 'Live load', 'Residual stress']

## 2. EXPERIMENT

# Model & Algorithm Selection (5 x 2 Nested Cross-Validation)




### Inner Loop

- Hyperparameter tuning
- Evaluate hyper parameters
- Find best parameter setting & Accuracy

### Outer loop

- Best setting from inner loop
- Evaluate it on the test fold
- Total 5 models & 5 Accuracy
- 1 Average Accuracy
- Choose the best model

### Repeat 3 times (*k*NN, DT, RF)

## 2. EXPERIMENT

# Model & Algorithm Selection (5 x 2 Nested Cross-Validation)

| Algorithm | Classifiers | Hyperparameter Grid |
|---|---|---|
| ***k*NN** | KNeighborsClassifier() | `[{'n_neighbors': list(range(3,15)), 'p': [1, 2], #1: Manhattan, 2: euclidean 'algorithm':['auto', 'kd_tree', 'ball_tree']}]` |
| **Decision Tree** | DecisionTreeClassifier(random_state=123 | `[{'max_depth': list(range(1, 20)) + [None], 'criterion': ['gini', 'entropy'], 'min_samples_split': [2, 3, 4]}]` |
| **Random Forest** | RandomForestClassifier(random_state=123, n_estimators = 1000) | `[{'max_depth': [1, 3, 11, 13, None], 'min_samples_split': [2, 3]}]` |

# Model & Algorithm Selection (5 x 2 Nested Cross-Validation)

| Algorithm | Chosen Hyperparameter | Average Accuracy |
|---|---|---|
| ***k*NN** | algorithm:auto<br>n_neighbors:15<br>p: 2 | 92.80% +/- 0.28 |
| **Decision Tree** | criterion:entropy<br>max_depth:15<br>min_samples_split:3 | 95.19% +/- 0.28 |
| **Random Forests** | max_depth':None<br>min_samples_split:3 | 97.43% +/- 0.11 |

**Best Algorithm:**

Random Forest (DT: 95.19% < **RF: 97.43%**)

However, the difference is **not** significant

Need to Check

k- Cross-Validation (k=10)

## 2. EXPERIMENT

# Model & Algorithm Selection (10 fold Cross-Validation)

**Purpose**

- Check if random forest is the best algorithm
- Model selection for random forest

| Algorithm | Best Parameters | Best CV Accuracy |
|---|---|---|
| Decision Tree | {'criterion': 'entropy', 'max_depth': 15, 'min_samples_split': 4} | 95.43% |
| Random Forest | {'max_depth': None, 'min_samples_split': 2} | 97.54% |

Still higher than decision tree

Best algorithm: Random Forest

**Chosen Model for prediction of failure in steel frames:**

Random Forest with hyperparameters {'max_depth':None, 'min_samples_split':2}

2. EXPERIMENT

# Evaluation

The best model: Random Forest {'max_depth': None, 'min_samples_split': 2, n_estimators: 1000}

1) **0.632 Bootstrap**
   - n_split: 200
   - Run Bootstrapping on the training set
   - Result
     - Mean Bootstrap score: 98.46 %
     - 95% Confidence interval: [98.32, 98.60]

2) **Fit on the training set & evaluate on the test set**
   - Result
     - Training Accuracy: 100.00%
     - Test Accuracy: 97.77%

## 2. EXPERIMENT

# Results: *k*NN

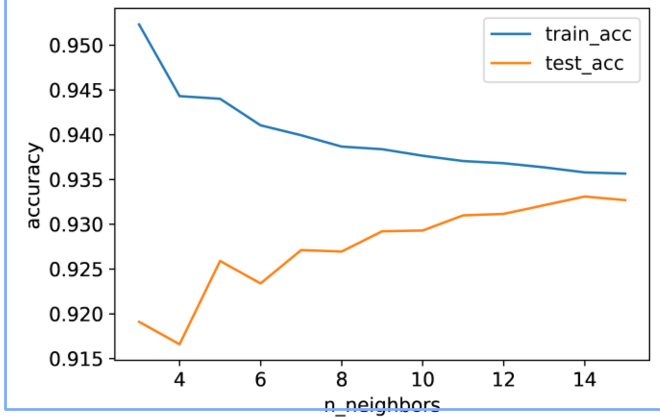
1) **Best model from nested CV of data_1:**

{'algorithm':'auto', 'n_neighbors':15, 'p':2}

2) **Test set accuracy** for the whole training data set with PCA: 93.26%

3) **Plot**

neighbors_settings = range(3,16)




n_neighbors':15; accuracy= 93.26%

n_neighbors':14; accuracy= 93.30%

## 2. EXPERIMENT

# Results: Decision Tree

### 1) Best model from nested CV of data_1:

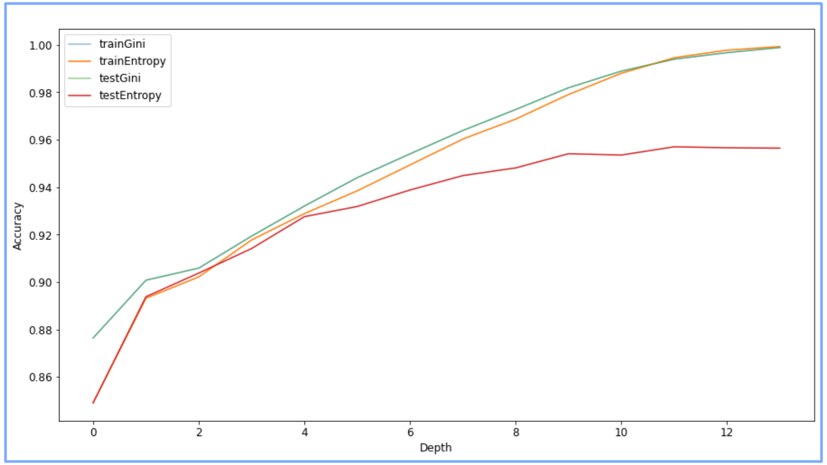
- Best model
  => {'criterion': 'entropy', 'max_depth': 15, 'min_samples_split': 3}
- Accuracy: 95.19%

### 2) By using 10-CV again and fit to the training data set,

- The Best 10-CV accuracy: 95.43%

### 3) Plot

{Plot_entropy vs gini; Depth}

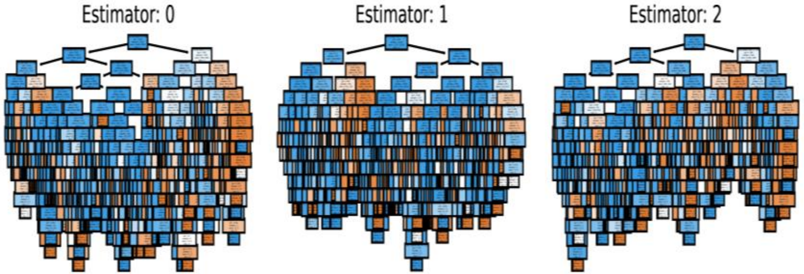



Optimize max_depth =15

## 2. EXPERIMENT

# Results: Random Forests

1) **Best model from Nested CV of data_1:**
   - best model => {'max_depth':None, 'min_samples_split':3}
   - Accuracy: 97.43%
2) **By using 10-CV again and fit to the training data set,**
   - The best 10-CV accuracy 97.54% with the same hyperparameter setting



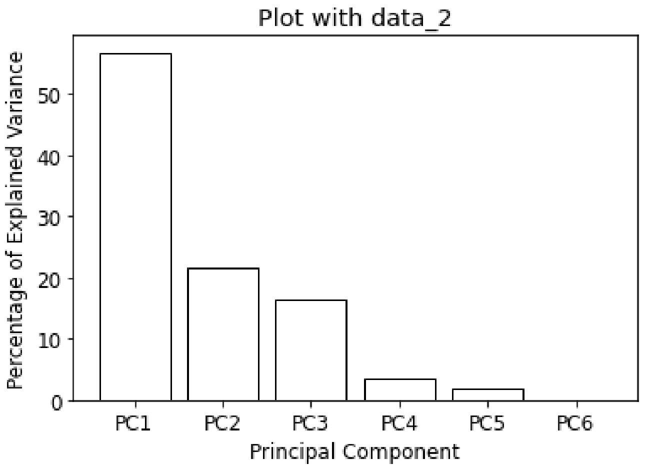
## 2. EXPERIMENT

# Application of Different Dataset (data_2)

### Feature Selection (PCA + SFS) only for *k*NN

### Principal Component Analysis (PCA)

- Explained Variance




### Sequential Feature Selector (SFS)

```
{1: {'feature_idx': (1,),
  'cv_scores': array([0.96951022]),
  'avg_score': 0.9695102173207915,
  'feature_names': ('1',)},
 2: {'feature_idx': (1, 2),
  'cv_scores': array([0.97948427]),
  'avg_score': 0.979484268569575,
  'feature_names': ('1', '2')},
 3: {'feature_idx': (0, 1, 2),
  'cv_scores': array([0.98617418]),
  'avg_score': 0.9861741809925397,
  'feature_names': ('0', '1', '2')}}
```

- **Selected features:**

['Dead load', 'Live load', 'Yield strength']

## 2. EXPERIMENT

# Application of Different Dataset (data_2)

### *k*NN

From data_1:

algorithm: auto

n_neighbors: 15

p: 2 (Euclidean)

10-fold Accuracy: 97.98%

Test Accuracy: 97.95%

### Decision Tree

From data_1:

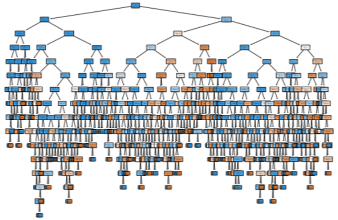
criterion: entropy

max_depth: 15

min_samples_split: 3

10-fold Accuracy: 98.13%

Test Accuracy: 98.09%



### Random Forest

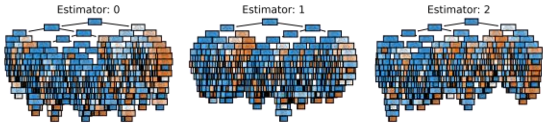
From data_1:

'max_depth': None,

'min_samples_split': 3

10-fold Accuracy: 97.99%

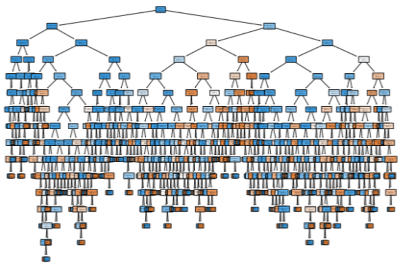
Test Accuracy: 98.84%



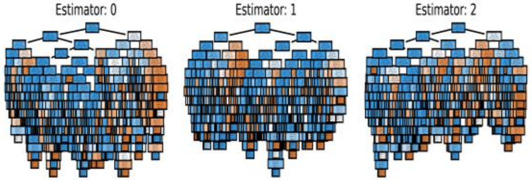
# 3. Discussion & Conclusion

- **Result: data_1**
  - Best Algorithm: **Random Forest** (whole dataset: 97.54%)
  - All three models exceed the target accuracy (93%)
- **Experiment with the three models**: weaknesses/shortcomings
  - ***k*NN**: Difficult to Visualize (3 features: 3 Dimensions)
  - **Decision Tree & Random forest**:
    - Plot: Trees are too deep & complex
    - Difficult to interpret

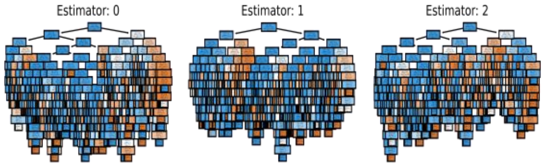
D.T(Data 2)



R.F(Data 1)




R.F(Data 2)



# 3. Discussion & Conclusion

| Algorithm | data_1 Accuracy | data_2 Accuracy |
|---|---|---|
| ***k*NN**:<br>algorithm: auto<br>n_neighbors: 15<br>p: 2 | 93.26% | 97.95% |
| **Decision Tree:**<br>criterion: entropy<br>max_depth: 15<br>min_samples_split: 3 | 95.36% | 98.09% |
| **Random Forests**<br>max_depth: None<br>min_samples_split: 3 | 97.77% | 98.84% |

3. **Applying the three models to data_1 & data_2**

   i. data_1 ACC < data_2 ACC
   ii. All test ACC above target( 93%)
   **iii.** **Random Forests** is the best algorithm for both data_1 & data_2
   iv. The best model can be used to the steel frames with different failure modes.

# Thanks!

**Any questions?**

hyeyoung.koh@wisc.edu

ypark258@wisc.edu

jyang633@wisc.edu