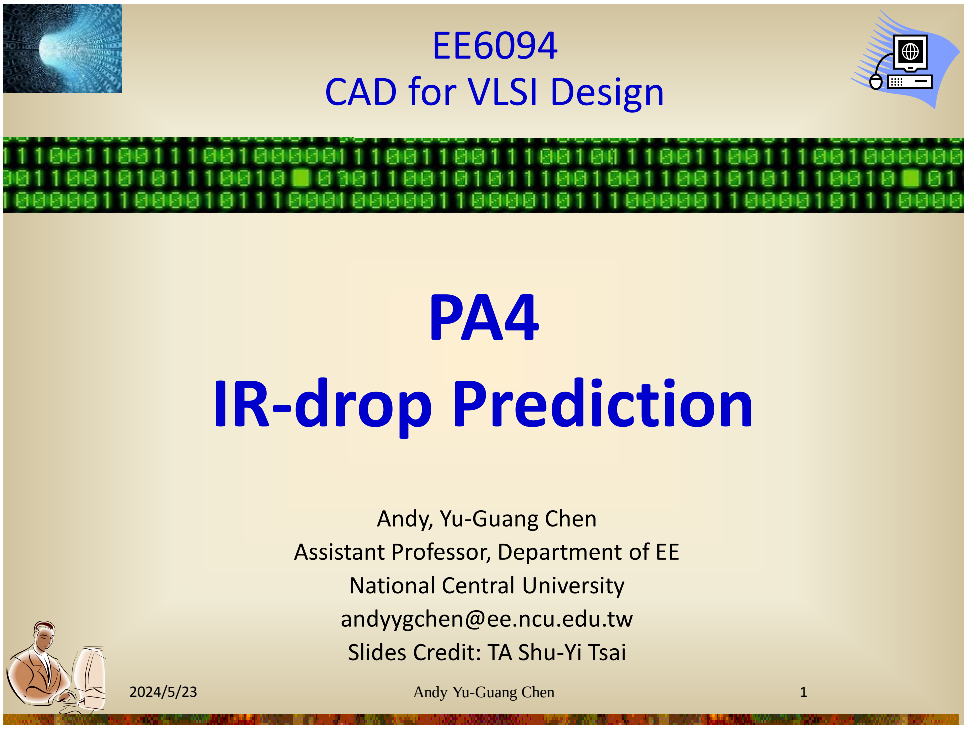
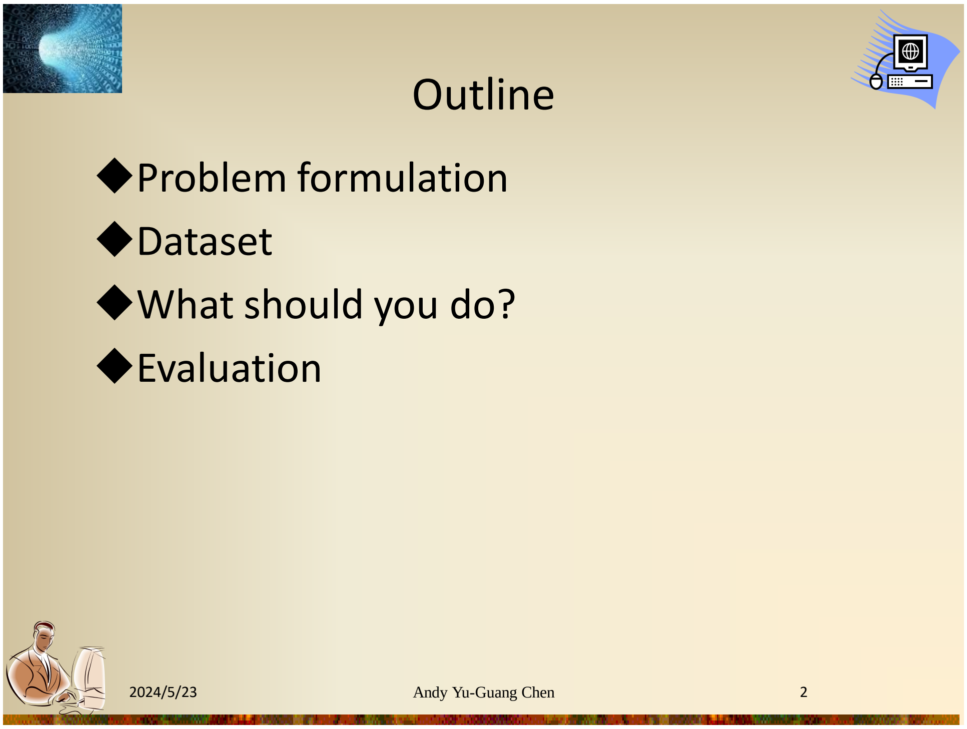


EE6094
CAD for VLSI Design

PA4
IR-drop Prediction

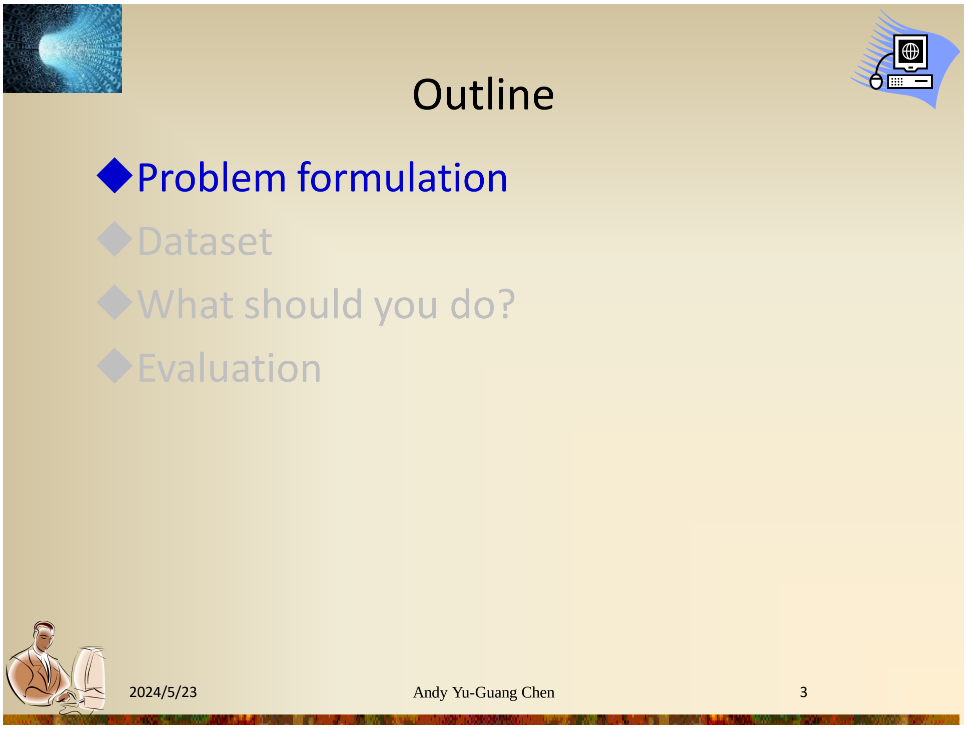
Andy, Yu-Guang Chen
Assistant Professor, Department of EE
National Central University
andyygchen@ee.ncu.edu.tw

Slides Credit: TA Shu-Yi Tsai



Outline

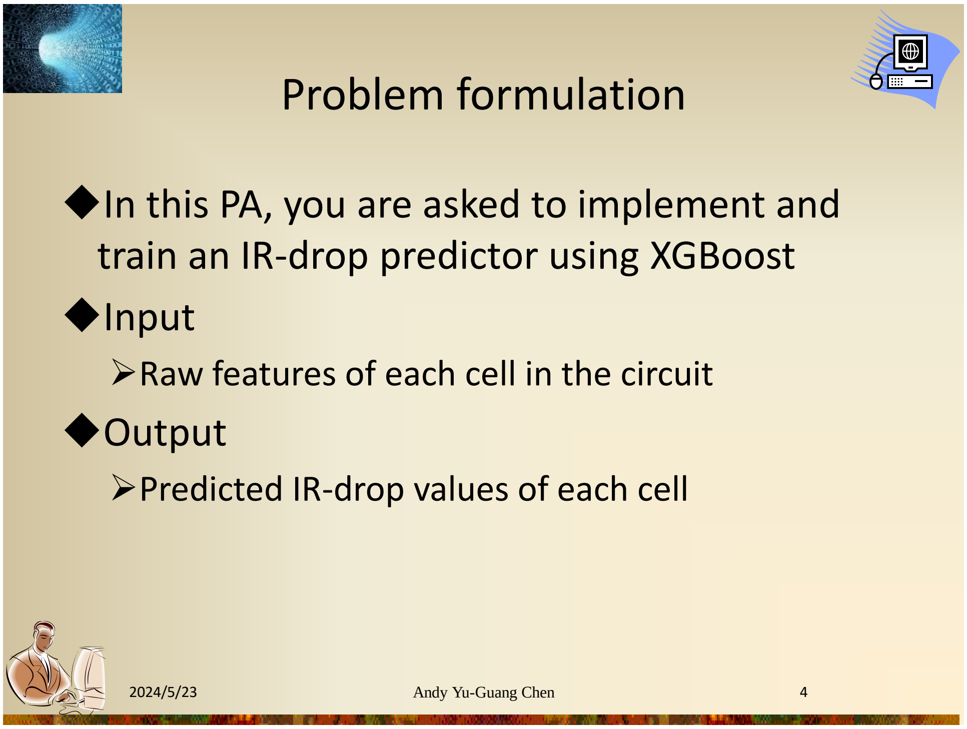
- ◆Problem formulation
- ◆Dataset
- ◆What should you do?
- ◆Evaluation



Outline

- Problem formulation
- Dataset
- What should you do?
- Evaluation

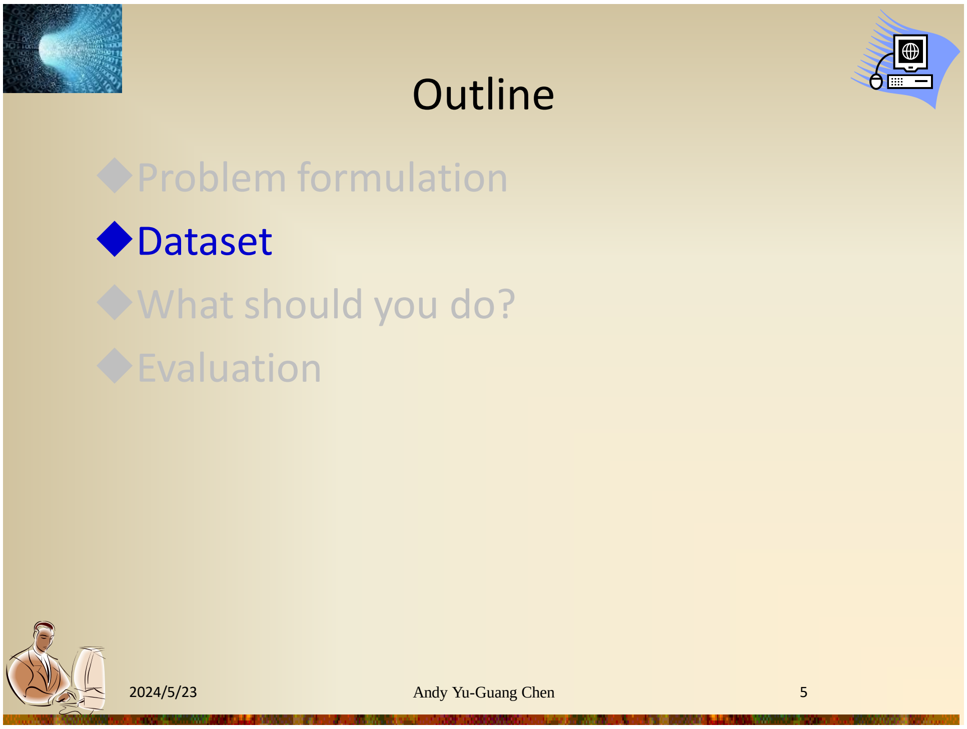
3



Problem formulation

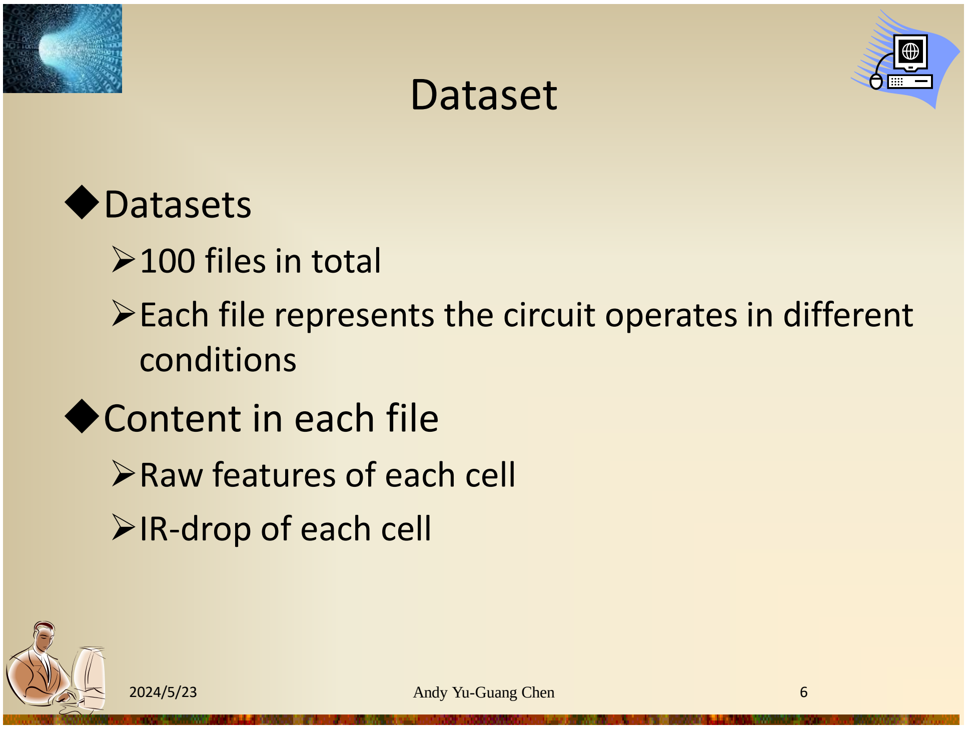
- In this PA, you are asked to implement and train an IR-drop predictor using XGBoost
- Input
  - Raw features of each cell in the circuit
- Output
  - Predicted IR-drop values of each cell

4



Outline

- Problem formulation
- Dataset
- What should you do?
- Evaluation

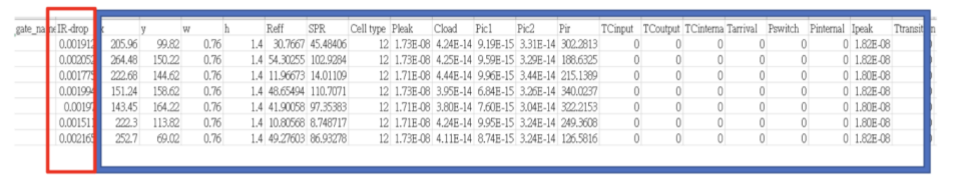


Dataset

- Datasets
  - 100 files in total
  - Each file represents the circuit operates in different conditions
- Content in each file
  - Raw features of each cell
  - IR-drop of each cell

Dataset

- ◆File format
  - ➢Each row contains the information of a cell
  - ➢Each column represents a feature of all cells



gate_na	IR-drop	x	y	w	h	Reff	SPR	Cell type	Pleak	Cload	Pic1	Pic2	Pir	TCinput	TCoutput	TCinterna	Tarrival	Pswitch	Pinternal	Ipeak	Ttransitin
	0.001912	205.96	99.82	0.76	1.4	30.7667	45.48406	12	1.73E-08	4.24E-14	9.19E-15	3.31E-14	302.2813	0	0	0	0	0	0	1.82E-08	0
	0.002052	264.48	150.22	0.76	1.4	54.30255	102.9284	12	1.73E-08	4.25E-14	9.59E-15	3.29E-14	188.6325	0	0	0	0	0	0	1.82E-08	0
	0.001775	222.68	144.62	0.76	1.4	11.96673	14.01109	12	1.71E-08	4.44E-14	9.96E-15	3.44E-14	215.1389	0	0	0	0	0	0	1.80E-08	0
	0.001994	151.24	158.62	0.76	1.4	48.65494	110.7071	12	1.73E-08	3.95E-14	6.84E-15	3.26E-14	340.0237	0	0	0	0	0	0	1.82E-08	0
	0.00197	143.45	164.22	0.76	1.4	41.90058	97.35383	12	1.71E-08	3.80E-14	7.60E-15	3.04E-14	322.2153	0	0	0	0	0	0	1.80E-08	0
	0.001511	222.3	113.82	0.76	1.4	10.80568	8.748717	12	1.71E-08	4.24E-14	9.95E-15	3.24E-14	249.3608	0	0	0	0	0	0	1.80E-08	0
	0.002165	252.7	69.02	0.76	1.4	49.27603	86.93278	12	1.73E-08	4.11E-14	8.74E-15	3.24E-14	126.5816	0	0	0	0	0	0	1.82E-08	0

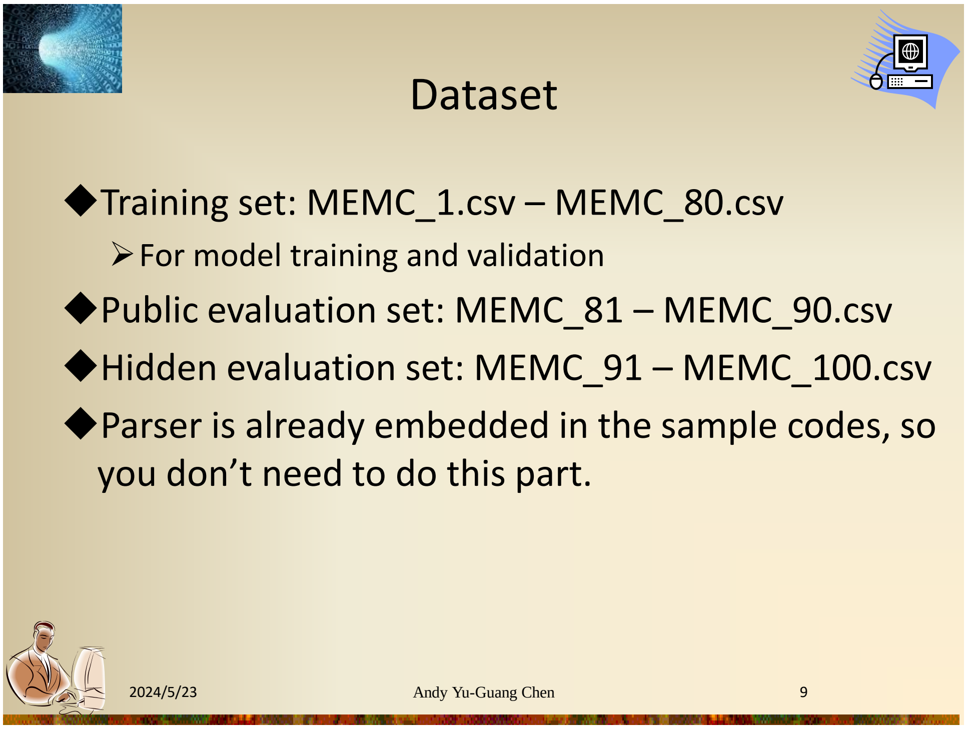
IR drop of each cell

Feature of each cell

Dataset

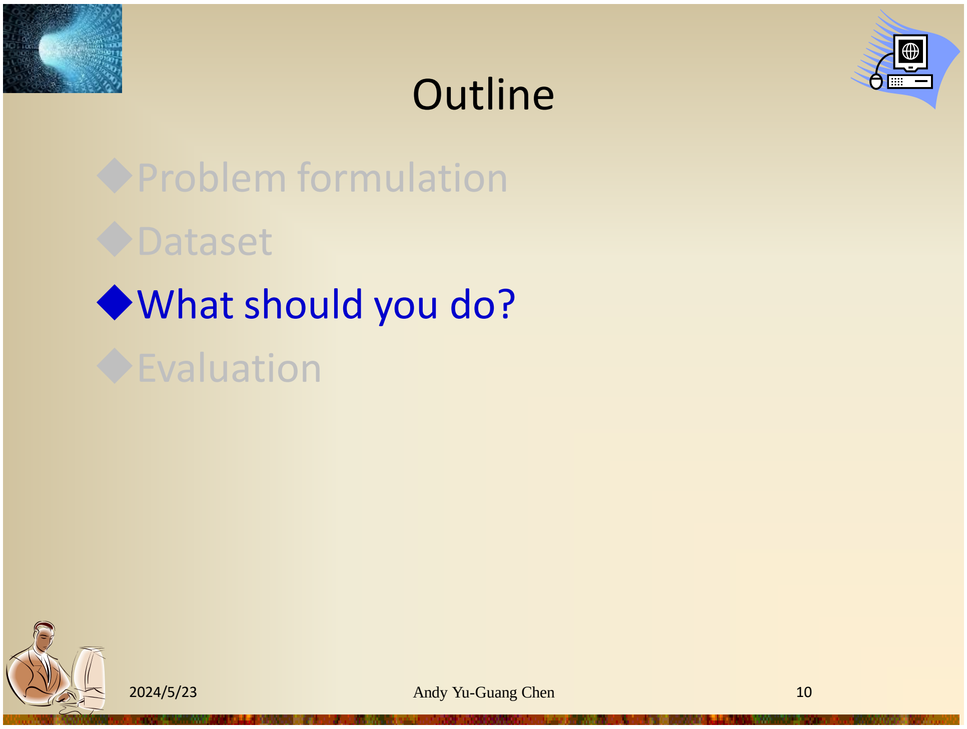
- ◆Descriptions of each feature

Feature	Description	Feature	Description
x, y	Physical location (coordinate)	TC_{input}	Toggle counts of input
w, h	Width, height (dimension)	TC_{output}	Toggle counts of output
R_{eff}	Effective resistance	$TC_{intertal}$	Toggle counts of internal connection
SPR	Shortest path resistance	$T_{arrival}$	Minimum arrival time
$Cell\ type$	Cell type	$P_{internal}$	Internal power
P_{leak}	Leakage power	P_{switch}	Switching power
C_{load}	Loading capacitance	$T_{transition}$	Transition time
Pi_{c1}, Pi_{c2}, Pi_{r}	Equivalent π model	I_{peak}	Peak current



Dataset

- Training set: MEMC_1.csv – MEMC_80.csv
  - For model training and validation
- Public evaluation set: MEMC_81 – MEMC_90.csv
- Hidden evaluation set: MEMC_91 – MEMC_100.csv
- Parser is already embedded in the sample codes, so you don't need to do this part.

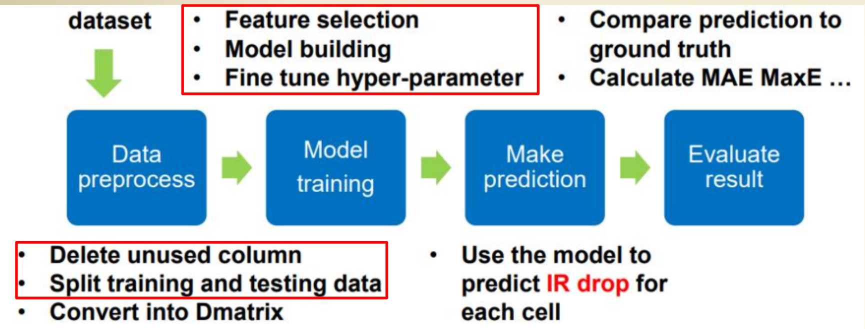


Outline

- Problem formulation
- Dataset
- What should you do?
- Evaluation

What should you do?

- ◆IR-drop prediction flow
  - ➢Most of the part are done, you only need to finish some TODOs



dataset

- Feature selection
- Model building
- Fine tune hyper-parameter

- Compare prediction to ground truth
- Calculate MAE MaxE …

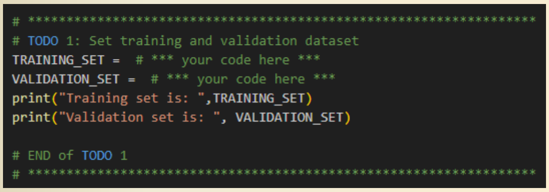
Data preprocess → Model training → Make prediction → Evaluate result

- Delete unused column
- Split training and testing data
- Convert into Dmatrix

- Use the model to predict IR drop for each cell

What should you do?

- ◆TODO 1: set training and validation set
- ◆Example:
  - ➢TRAINING_SET = np.arange(10)
    (MEMC_1.csv to MEMC_10.csv are for training)
  - ➢VALIDATION_SET = np.array([20])
    (MEMC_20.csv is for validation)

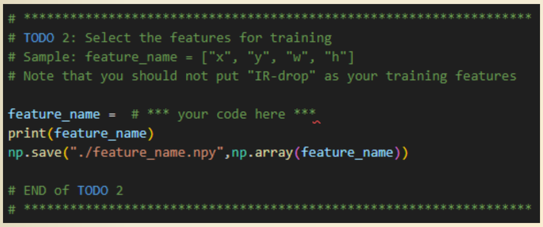


```
# ******************************************************************
# TODO 1: Set training and validation dataset
TRAINING_SET =  # *** your code here ***
VALIDATION_SET =  # *** your code here ***
print("Training set is: ",TRAINING_SET)
print("Validation set is: ", VALIDATION_SET)

# END of TODO 1
# ******************************************************************
```

What should you do?

- ◆TODO 2: select the features
  - ➢Note that “IR-drop” should not be selected as your feature, or you will get no points



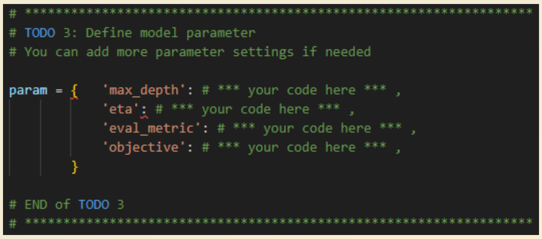
```
# ******************************************************************
# TODO 2: Select the features for training
# Sample: feature_name = ["x", "y", "w", "h"]
# Note that you should not put "IR-drop" as your training features

feature_name =  # *** your code here ***
print(feature_name)
np.save("./feature_name.npy",np.array(feature_name))

# END of TODO 2
# ******************************************************************
```

What should you do?

- ◆TODO 3: set the hyper parameters of XGBoost
- ◆Example:
  - ➢max_depth: 1
  - ➢eta: 0.3 (learning rate of xgboost)
  - ➢eval_metric: mae
  - ➢Objective: reg:squarederror



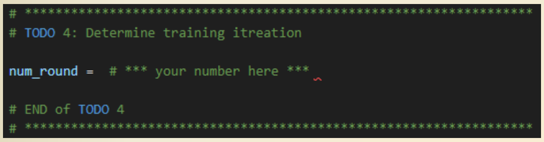
```
# ******************************************************************
# TODO 3: Define model parameter
# You can add more parameter settings if needed

param = {   'max_depth': # *** your code here *** ,
            'eta': # *** your code here *** ,
            'eval_metric': # *** your code here *** ,
            'objective': # *** your code here *** ,
        }

# END of TODO 3
# ******************************************************************
```

What should you do?

- ◆TODO 4: set training iteration
- ◆Example:
  - ➢num_round = 3



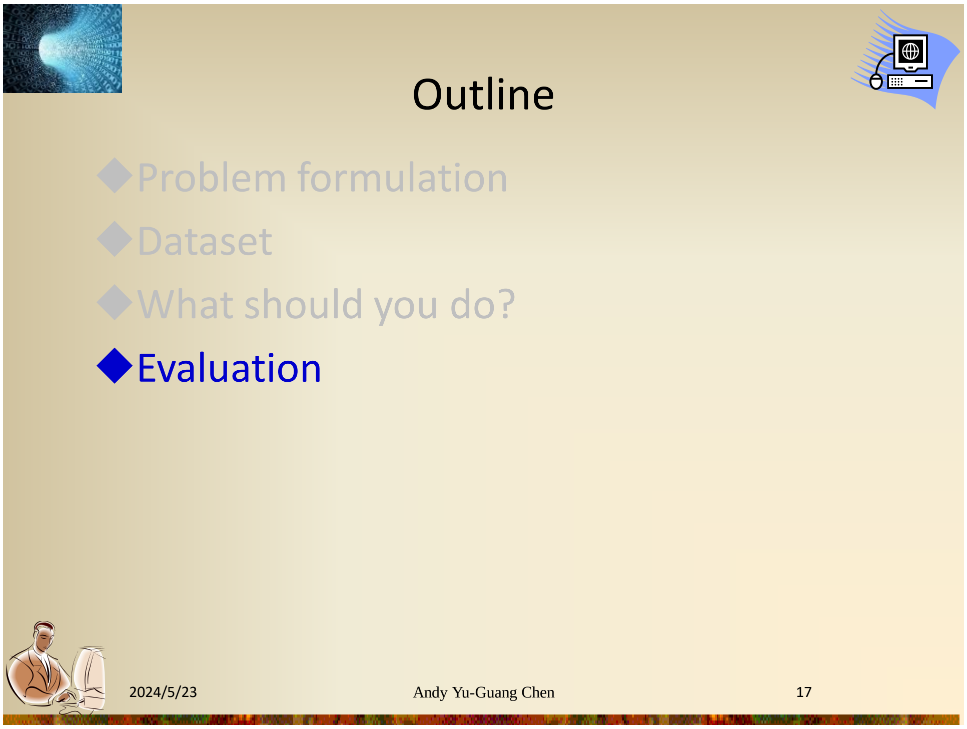
```
# ******************************************************************
# TODO 4: Determine training itreation

num_round =  # *** your number here ***

# END of TODO 4
# ******************************************************************
```

What should you do?

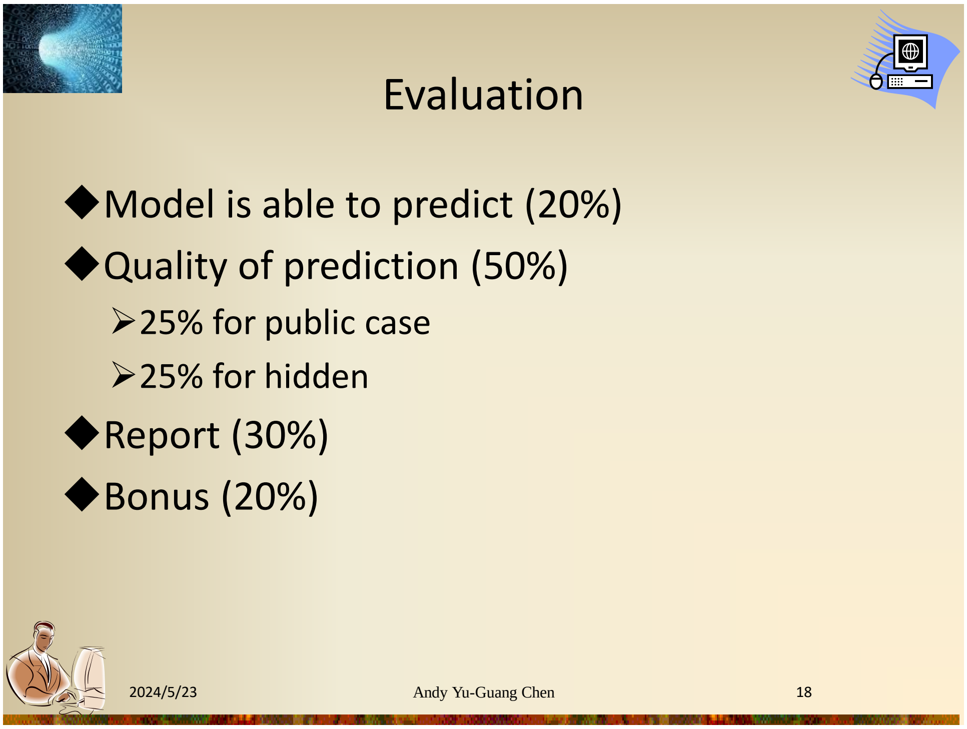
- ◆What should you do in PA4?
  1. Create a folder for PA4 and upload the sample codes
  2. Finish TODOs in Sample_training.py
  3. Run Sample_training.py for model training
  4. Run Sample_evaluation.py to get the result
     - Note that you don’t need modify this code
  5. Fine-tune your model



Outline

- Problem formulation
- Dataset
- What should you do?
- Evaluation

17



Evaluation

- Model is able to predict (20%)
- Quality of prediction (50%)
  - 25% for public case
  - 25% for hidden
- Report (30%)
- Bonus (20%)

18

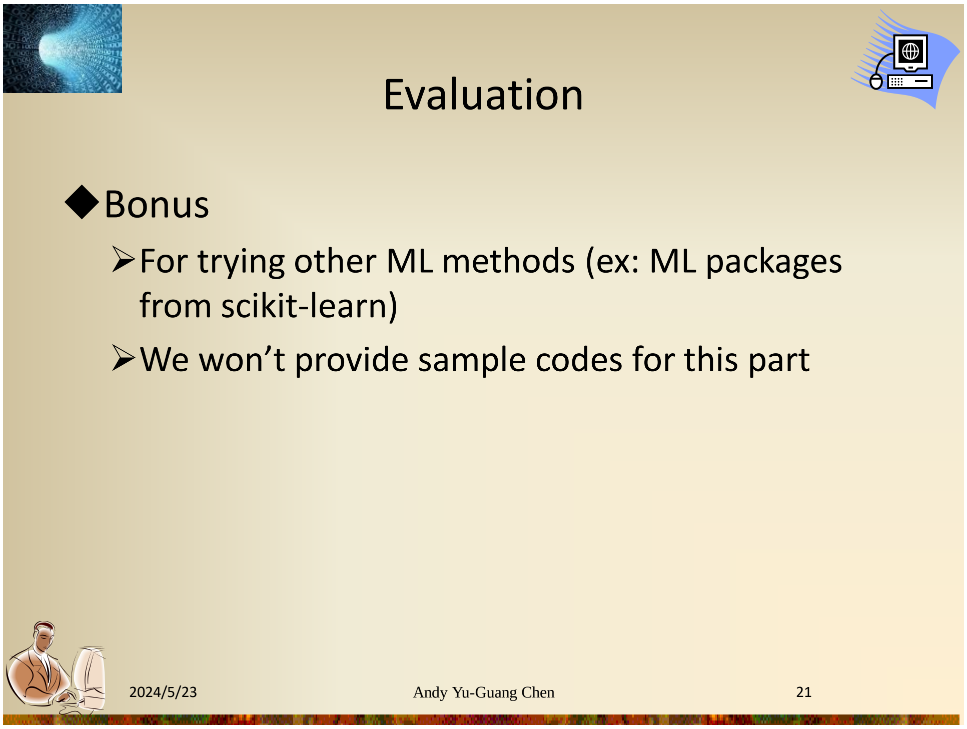
Evaluation

- ◆The quality will be evaluated by 4 metrics
  - ➢Mean absolute error (MAE)
  - ➢Max error (MaxE)
  - ➢Correlation Coefficient (CC)
  - ➢Normalized root mean squared error (NRMSE)
- ◆Public and hidden case will be calculated separately

$$Quality\ (25\%) = 5 + \sum_{4\ metrics} \left(1 - \frac{Difference\ between\ your\ result\ and\ best\ result}{Difference\ between\ best\ and\ worse\ result}\right) * 5$$

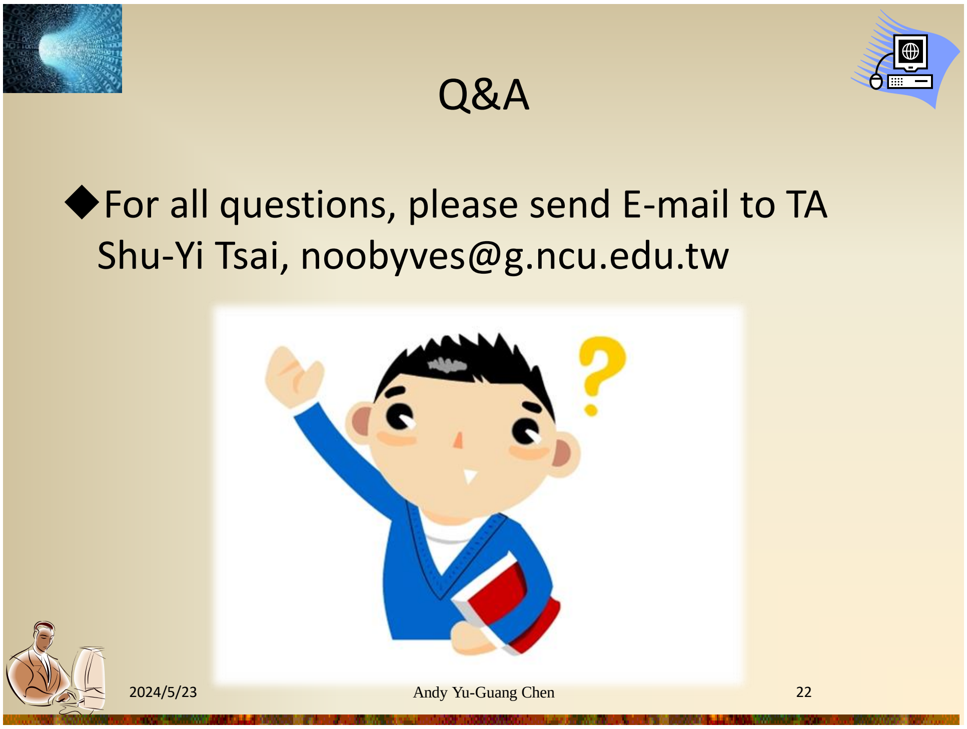
Evaluation

- ◆Report
  - ➢You should at least include:
    - How you finish the 4 TODO spaces
    - Evaluation result (output of Sample_evaluation.py)
    - Hardness
    - Suggestion?
  - ➢We don’t restrict the report format and length
  - ➢English version is a plus
  - ➢The grading of the report will compare yours with others



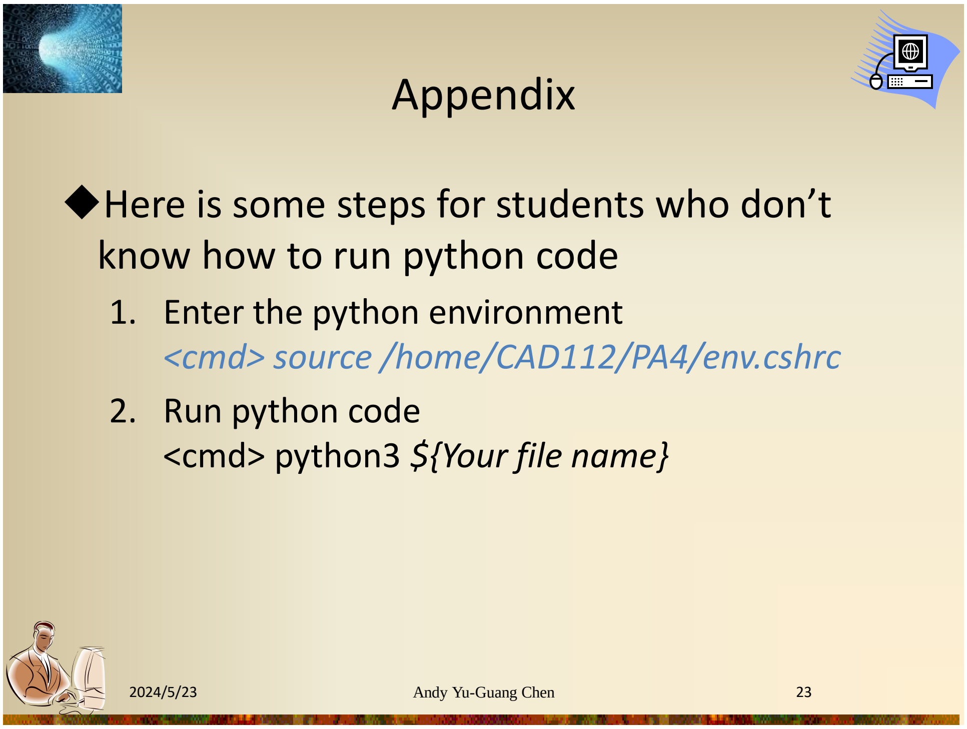
Evaluation

- Bonus
  - For trying other ML methods (ex: ML packages from scikit-learn)
  - We won’t provide sample codes for this part



Q&A

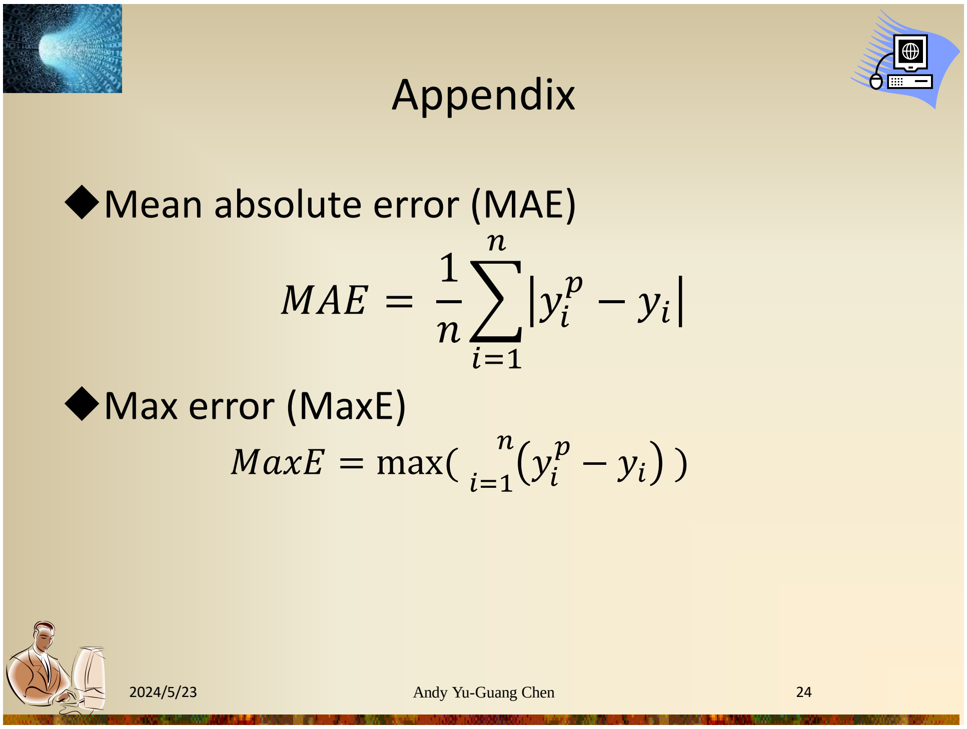
- For all questions, please send E-mail to TA Shu-Yi Tsai, noobyves@g.ncu.edu.tw



Appendix

- Here is some steps for students who don’t know how to run python code
  1. Enter the python environment
     <cmd> source /home/CAD112/PA4/env.cshrc
  2. Run python code
     <cmd> python3 *${Your file name}*

23



Appendix

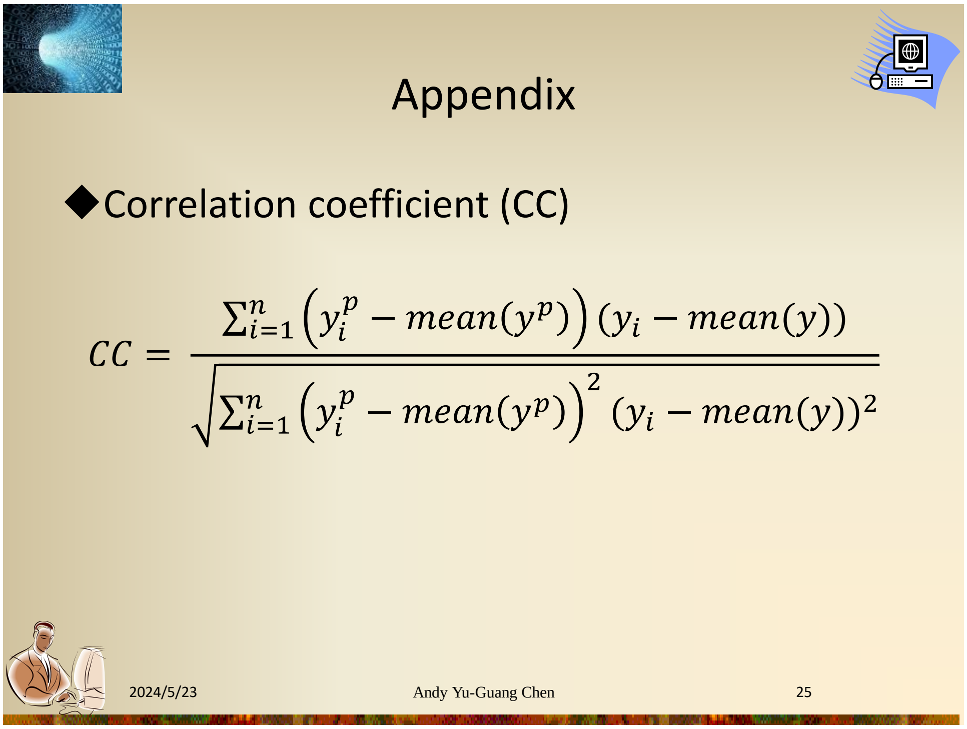
- Mean absolute error (MAE)

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i^p - y_i\right|$$

- Max error (MaxE)

$$MaxE = \max({}_{i=1}^{n}(y_i^p - y_i))$$

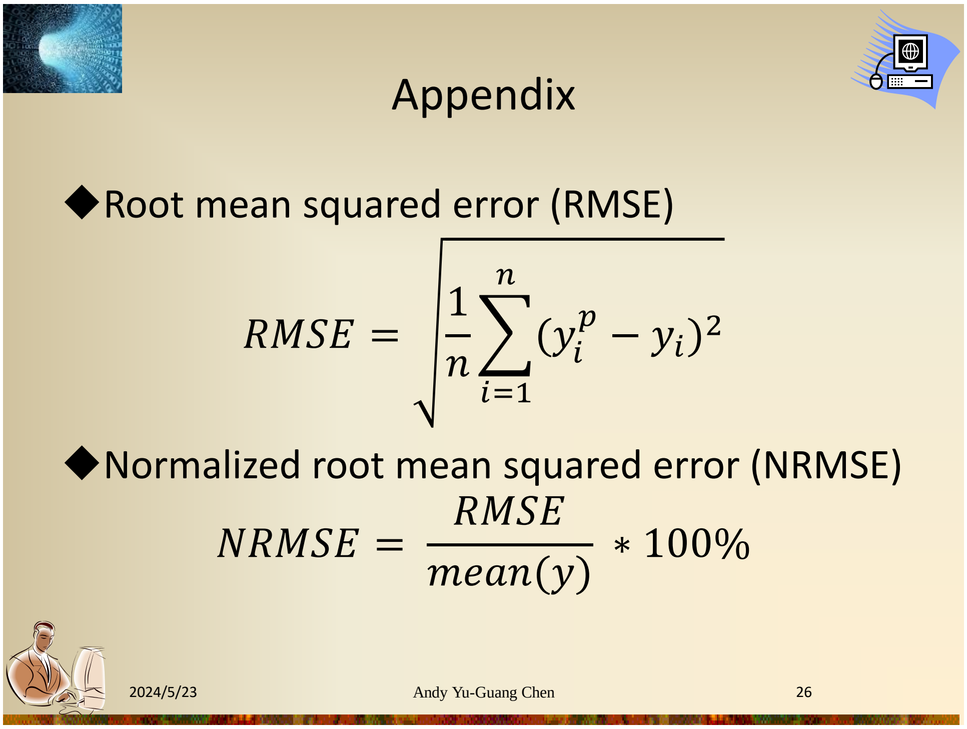
24



Appendix

- Correlation coefficient (CC)

$$CC = \frac{\sum_{i=1}^{n} \left(y_i^p - mean(y^p)\right)(y_i - mean(y))}{\sqrt{\sum_{i=1}^{n} \left(y_i^p - mean(y^p)\right)^2 (y_i - mean(y))^2}}$$



Appendix

- Root mean squared error (RMSE)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i^p - y_i)^2}$$

- Normalized root mean squared error (NRMSE)

$$NRMSE = \frac{RMSE}{mean(y)} * 100\%$$

Andy, Yu-Guang Chen
Assistant Professor, Department of EE, NCU
Email: andyygchen@ee.ncu.edu.tw
FB: Yu-Guang Chen
IG: ncu.eda.andy
Google account: andyygchen.ncu