

학력

학부 : 부산대학교 화공생명공학부 졸업 (2008.03 ~ 2014.02)

석사: KAIST 생명화학공학과 졸업 (2014.03 ~ 2016.02)

박사: KAIST 생명화학공학과 박사과정 재학중 (2016.03 ~ 현재)

실험실: Molecular Simulation Laboratory (Prof. Jihan Kim)

세부 전공: Molecular simulation, computational chemistry,
Machine learning, deep learning


활동

KaggleKorea 페이스북 온라인 그룹(현재 약4,600명) 운영자

대전 캐글스터디(50명), 부산 캐글스터디(40명) 운영자



이유한



YouHan Lee

Ph.D student at KAIST
대전광역시, 대전광역시, 대한민국
Joined 2 years ago · last seen in the past day

Followers 195
Following 31

Competitions Expert

[Home](#)
[Competitions \(20\)](#)
[Kernels \(32\)](#)
[Discussion \(294\)](#)
[Datasets](#)
...
[Edit Profile](#)

Competitions Expert

Current Rank	Highest Rank
695 of 103,093	677

0
3
3

- Quick, Draw! Doodle Recog...
4 months ago-Top 2%
24th of 1316
- Microsoft Malware Prediction
a month ago-Top 2%
40th of 2426
- Elo Merchant Category Rec...
a month ago-Top 3%
86th of 4129

Kernels Expert

Current Rank	Highest Rank
37 of 89,782	36

4
5
16

- My EDA - I want to see all!
3 months ago
177 votes
- Simple quant features usin...
6 months ago
117 votes
- YH EDA - I want to see all!!
2 months ago
104 votes

Discussion Expert

Current Rank	Highest Rank
108 of 89,895	104

3
10
95

- My prize is always what I've...
6 months ago
18 votes
- 중요한 공지입니다. 확인하세...
3 months ago
17 votes
- Insight or Dodgy
6 months ago
13 votes

What is Kaggle?

캐글 as a company

- 2010년 설립된 빅데이터 솔루션 대회 플랫폼 회사
- 2017 년 3월에 구글에 인수

kaggle

캐글 as a community

- 현재 200만명의 회원 보유
- Data science, ML, DL 을 주제로 모인 community

kaggle

Competition - Data Race for 데이터 과학자!

기업, 정부기관, 단체, 연구소, 개인

**Dataset
With Prize**

kaggle

**Dataset & Prize
개발 환경(kernel)
커뮤니티(follow, discussion)**

전 세계 데이터 사이언티스트

Dataset - Data Playground for 데이터 과학자!

기업, 정부기관, 단체, 연구소, 개인

**Dataset
With or without Prize**

kaggle

**Dataset & Prize
개발 환경(kernel)
커뮤니티(follow, discussion)**

전 세계 데이터 사이언티스트

Dataset

2019/04/10 기준 캐글에 등록되어 있고,
다운받을 수 있는 데이터셋 숫자는
15,722 개

Kernel!

- 캐글에서 제공하는 가상환경.
 - 컴퓨터 수십, 수백 대 제공해 줍니다.
 - With GPU!!!
- 검증된 캐글러들이 자신이 분석한 것을 공유합니다.
 - 좋은 reference, 공부 자료!

Why do kaggle?

빅데이터 시대,
4차 산업시대
가장 중요한 것이
뭘까요?

데이터

결정

인간의 경험 직관

결정

Data-driven approach

Data-driven Approach

데이터 안에 뭐가 있길래?

Pattern

패턴

머신 러닝 ?

Pattern recognition

머신 러닝 ?

Make **general function(conditions)**
to obtain goal(minimize loss)

머신 러닝 ?

Learn **statistics(correlations)** between
feature vs feature/
feature vs target

데이터 사이언스,
머신러닝,
딥러닝에서
가장 중요한 것은?

DATA

Data

Data

Dataaaaaaaaaaaaaa!!!!!!

Your neural network is only as good as the data you feed it.

당신의 모델은,
당신이 input으로 준 것 만큼 좋다!

Garbage in, Garbage out!

이런 것을
공부하려면?

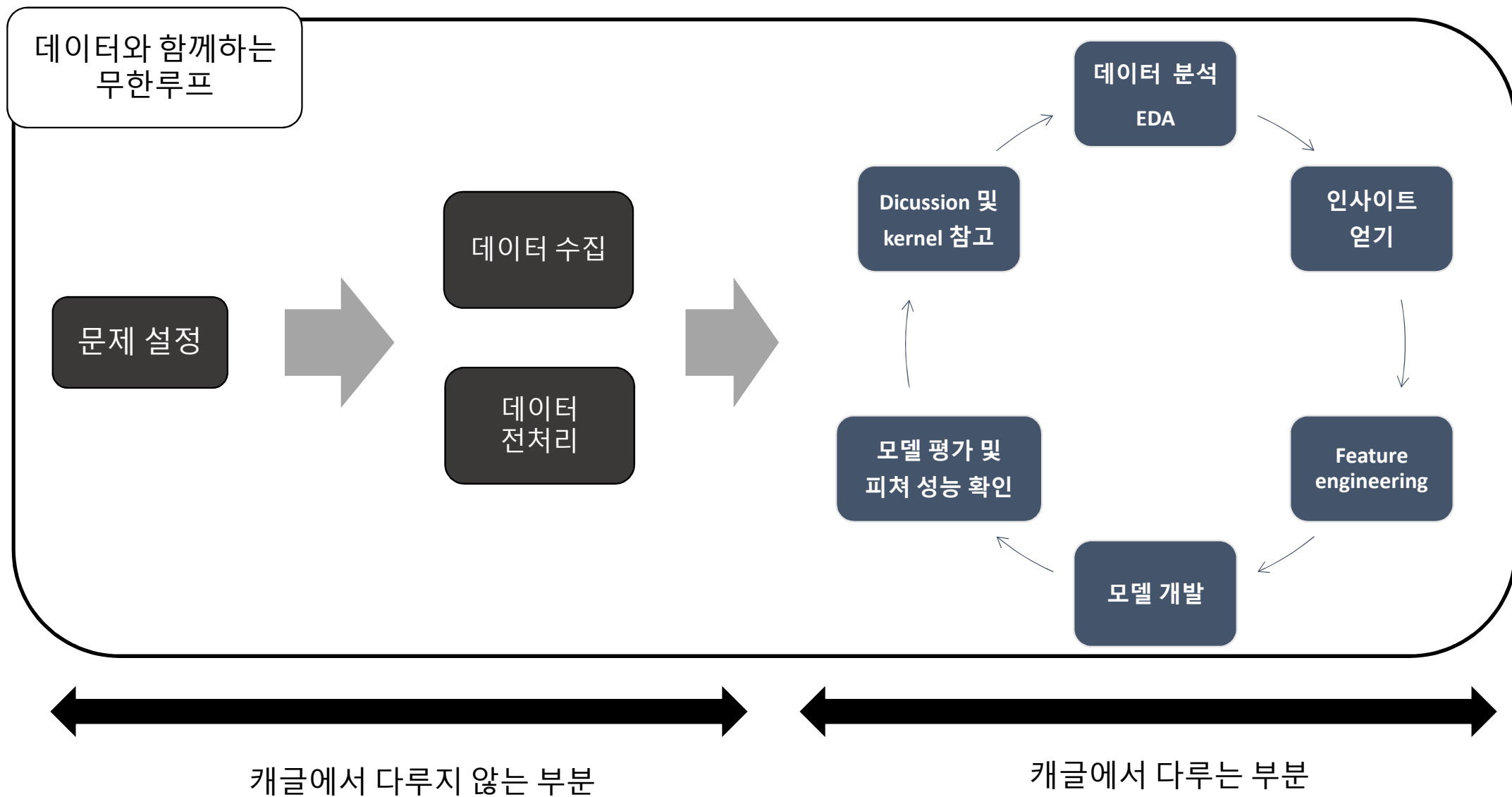
Data 와 사람(자료)이 많은
곳으로 가라!



캐글 코리아

Kaggle Korea

Non-Profit Facebook Group Community



머신러닝으로 할 수 있는 대부분의 문제 유형을 담고 있는 컴퍼티션들

지금까지 302개의
competition 이 치뤄짐.



머신러닝으로 풀 수 있는
대부분의 문제가 담겨있다

개인적 측면 – 경험 실력 FOR 정형 데이터

Porto: 고객이 내년에 자동차 보험금 청구를 할 것인가?

Home Credit: 고객이 앞으로 대출 상환을 할 것인가?

Costa rican: 고객의 소득 수준을 ML 로 구분하라

Elo: 거래 내역 데이터를 가지고, 고객 충성도를 예측하라

New York taxi: Taxi 탑승 시간을 예측하라

직방: 아파트 거래가격 예측하라

INFOCARE: 아파트 경매가격 예측해라

개인적 측면 – 경험 실력 FOR 정형 데이터

- Exploratory data analysis
 - Data visualization
 - Matplotlib, Seaborn, Plotly
 - Data mining
 - Pandas, numpy
- Feature engineering
 - Time series features
 - Categorical features
 - Numerical features
 - Aggregation features
 - Ratio features
 - Product features
- Data preparation
 - Data augmentation (imbalance)
 - Upsampling
 - Downsampling
 - SMOTE
- Model development
 - Sklearn
 - Linear model
 - Non-linear model
 - Tree-model
 - Not sklearn
 - Xgboost
 - Lightgbm
 - Catboost
 - LibFFM

개인적 측면 – 경험 실력 FOR 딥러닝

- Model evaluation
 - Various metrics
 - Accuracy
 - Precision
 - Recall
 - F1-score
 - Etc.
- Other technique
 - Machine learning pipeline
 - My pipeline code
 - Feature management

정형 데이터 위해 학습한 모델만,,,
천단위 이상으로 만들어 봤을 겁니다.

개인적 측면 – 경험 실력 FOR 딥러닝

Tensorflow: 30개 단어를 구분하는 AI 만들어라

Quora: 성실한, 불성실한 질문을 구분해내라

Doodle: 340개의 클래스 별 낙서를 ANN 으로 구분하라.

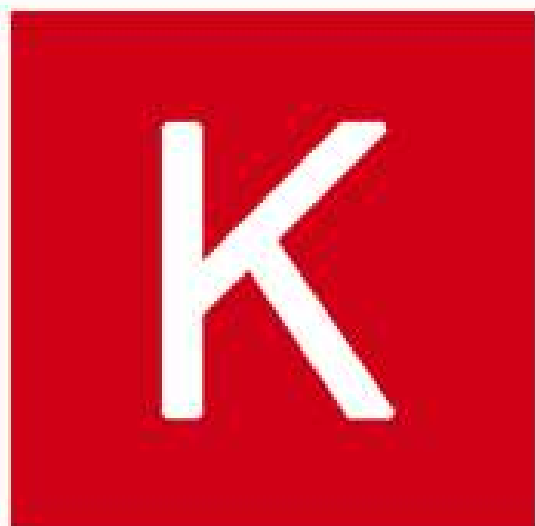
Protein: 28개의 클래스 별 Protein 을 ANN 으로 구분하라.

Airbus: 바다 위 배를 찍은 위성 사진에서 배의 위치를 찾아내라

Statoil: 바다 위 빙산과, 배를 구분하라

개인적 측면 – 경험 실력 FOR 딥러닝

Keras: The Python Deep Learning library



Keras

개인적 측면 – 경험 실력 FOR 딥러닝

- Model Development
 - Not pretrained
 - CNN
 - RNN
 - Simese network
 - Pretrained
 - Fine-tunning
- Learning technique
 - Cyclic learning
 - Generator
 - Data augmentation

딥러닝 모델만,
몇백개 이상 만든 거 같네요.

AI 사고 방식 탑재

문제 정하기

데이터 수집

데이터 분석

모델 만들기
(예측, 군집, 강화 학습)

Problem
Setting을
얼마나
잘 하느냐?

AI가 풀기 좋게
데이터를 어떻게
준비하느냐?

데이터 존재

문제 정하기

데이터 분석

모델 만들기
(예측, 군집, 강화 학습)

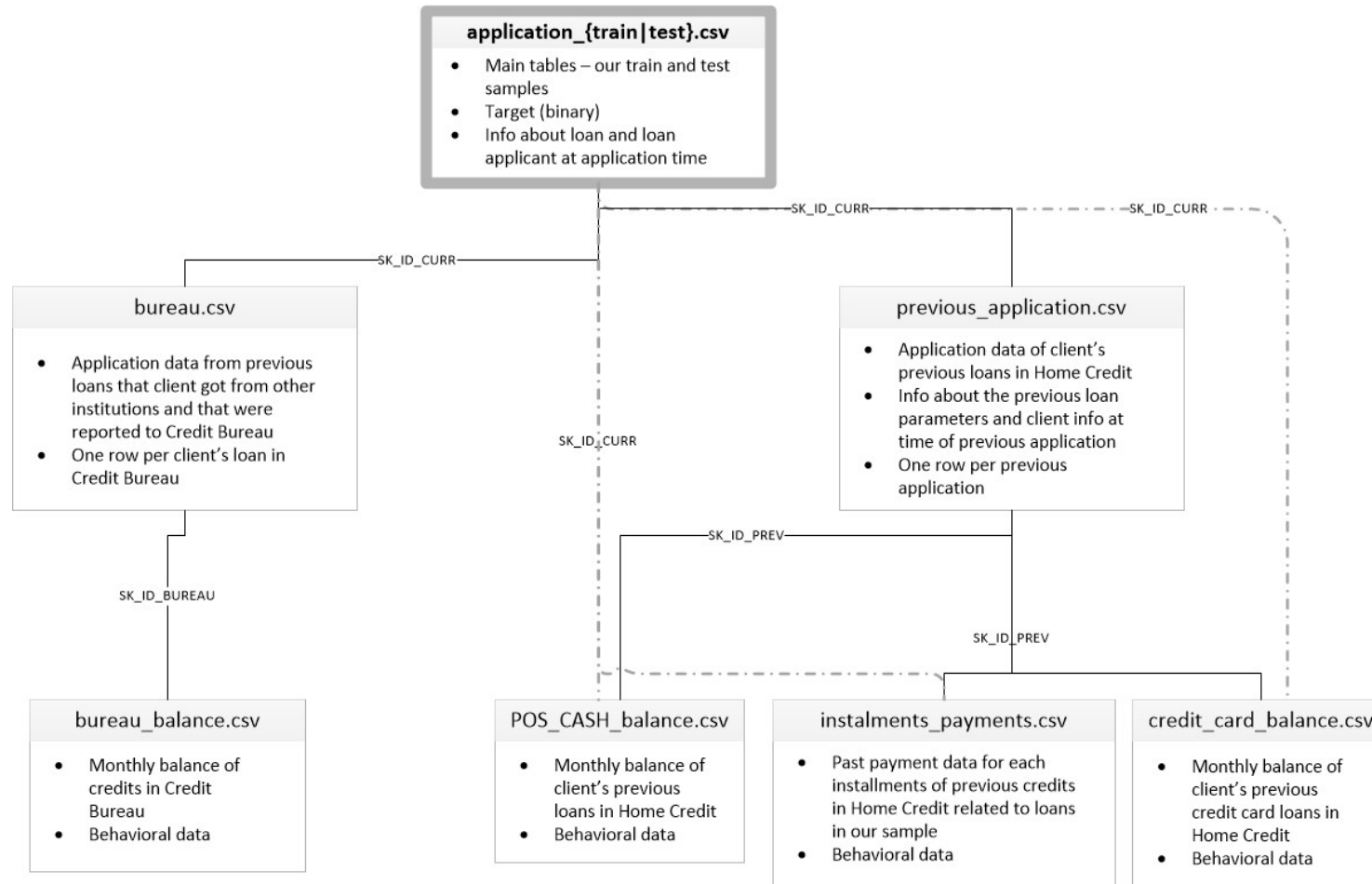
캐글 메달 Tip

Home Credit Default Risk competition - Timeline

- **August 22, 2018** - Entry deadline. You must accept the competition rules before this date in order to compete.
- **August 22, 2018** - Team Merger deadline. This is the last day participants may join or merge teams.
- **August 29, 2018** - Final submission deadline.

All deadlines are at 11:59 PM UTC on the corresponding day unless otherwise noted. The competition organizers reserve the right to update the contest timeline if they deem it necessary.

Home Credit Default Risk competition - Data



무엇부터
해야 할까요?

Home Credit Default Risk competition – Data description

- bureau_balance.csv
 - Monthly balances of previous credits in Credit Bureau.
 - This table has one row for each month of history of every previous credit reported to Credit Bureau – i.e the table has (#loans in sample * # of relative previous credits * # of months where we have some history observable for the previous credits) rows.
- POS_CASH_balance.csv
 - Monthly balance snapshots of previous POS (point of sales) and cash loans that the applicant had with Home Credit.
 - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (#loans in sample * # of relative previous credits * # of months in which we have some history observable for the previous credits) rows.
- credit_card_balance.csv
 - Monthly balance snapshots of previous credit cards that the applicant has with Home Credit.
 - This table has one row for each month of history of every previous credit in Home Credit (consumer credit and cash loans) related to loans in our sample – i.e. the table has (#loans in sample * # of relative previous credit cards * # of months where we have some history observable for the previous credit card) rows.
- previous_application.csv
 - All previous applications for Home Credit loans of clients who have loans in our sample.
 - There is one row for each previous application related to loans in our data sample.
- installments_payments.csv
 - Repayment history for the previously disbursed credits in Home Credit related to the loans in our sample.
 - There is a) one row for every payment that was made plus b) one row each for missed payment.
 - One row is equivalent to one payment of one installment OR one installment corresponding to one payment of one previous Home Credit credit related to loans in our sample.

이 컴퍼티션은 팀으로 참여하였습니다.

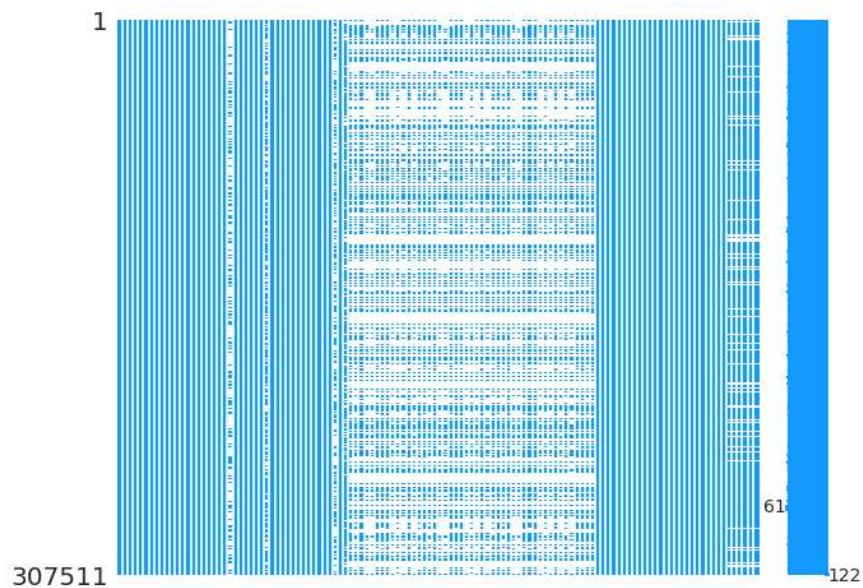
김연민님, 안병복님,
유용균님, 최성환님
그리고 저까지
총 5명이 함께 달렸습니다

1. Data check – Feature check

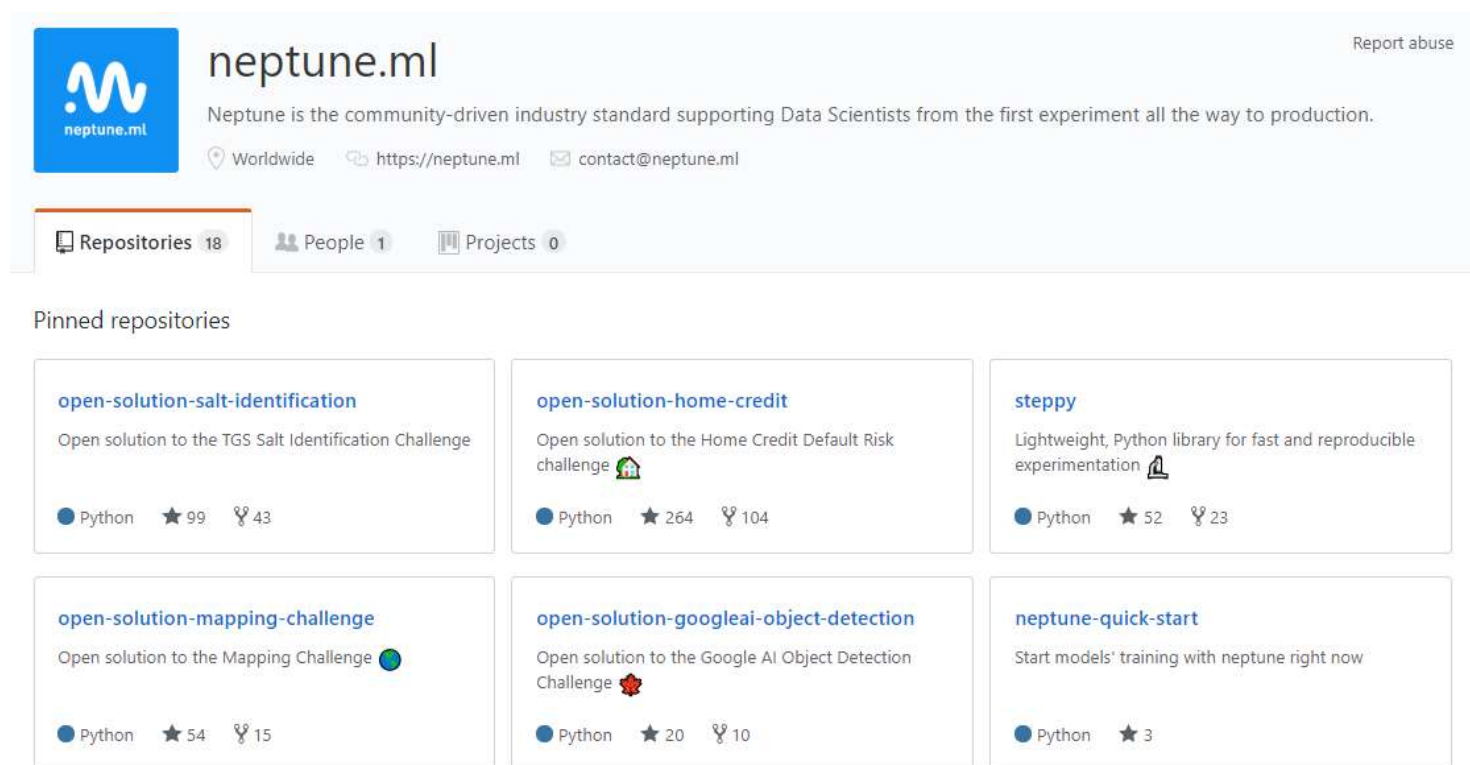
	role	level	dtype	response_rate
varname				
SK_ID_CURR	id	ordinal	int64	100.000000
TARGET	target	ordinal	int64	100.000000
NAME_CONTRACT_TYPE	input	categorical	object	100.000000
CODE_GENDER	input	categorical	object	100.000000
FLAG_OWN_CAR	input	categorical	object	100.000000
FLAG_OWN_REALTY	input	categorical	object	100.000000
CNT_CHILDREN	input	ordinal	int64	100.000000
AMT_INCOME_TOTAL	input	interval	float64	100.000000
AMT_CREDIT	input	interval	float64	100.000000
AMT_ANNUITY	input	interval	float64	99.996098
AMT_GOODS_PRICE	input	interval	float64	99.909597
NAME_TYPE_SUITE	input	categorical	object	99.579852
NAME_INCOME_TYPE	input	categorical	object	100.000000
NAME_EDUCATION_TYPE	input	categorical	object	100.000000
NAME_FAMILY_STATUS	input	categorical	object	100.000000
NAME_HOUSING_TYPE	input	categorical	object	100.000000
REGION_POPULATION_RELATIVE	input	interval	float64	100.000000
DAYS_BIRTH	input	ordinal	int64	100.000000
DAYS_EMPLOYED	input	ordinal	int64	100.000000
DAYS_REGISTRATION	input	interval	float64	100.000000
DAYS_ID_PUBLISH	input	ordinal	int64	100.000000
OWN_CAR_AGE	input	interval	float64	34.009190
FLAG_MOBIL	input	ordinal	int64	100.000000
FLAG_EMP_PHONE	input	ordinal	int64	100.000000
FLAG_WORK_PHONE	input	ordinal	int64	100.000000
FLAG_CONT_MOBILE	input	ordinal	int64	100.000000
FLAG_PHONE	input	ordinal	int64	100.000000
FLAG_EMAIL	input	ordinal	int64	100.000000
OCCUPATION_TYPE	input	categorical	object	68.654455
CNT_FAM_MEMBERS	input	interval	float64	99.999350

1. Data check – Null data check

	Total	Percent
COMMONAREA_MEDI	214865	69.872297
COMMONAREA_AVG	214865	69.872297
COMMONAREA_MODE	214865	69.872297
NONLIVINGAPARTMENTS_MODE	213514	69.432963
NONLIVINGAPARTMENTS_MEDI	213514	69.432963
NONLIVINGAPARTMENTS_AVG	213514	69.432963
FONDKAPREMONT_MODE	210295	68.386172
LIVINGAPARTMENTS_MEDI	210199	68.354953
LIVINGAPARTMENTS_MODE	210199	68.354953
LIVINGAPARTMENTS_AVG	210199	68.354953
FLOORSMIN_MEDI	208642	67.848630
FLOORSMIN_MODE	208642	67.848630
FLOORSMIN_AVG	208642	67.848630
YEARS_BUILD_MEDI	204488	66.497784
YEARS_BUILD_AVG	204488	66.497784
YEARS_BUILD_MODE	204488	66.497784
OWN_CAR_AGE	202929	65.990810
LANDAREA_MODE	182590	59.376738
LANDAREA_AVG	182590	59.376738
LANDAREA_MEDI	182590	59.376738



1. Data check – Outlier check



The screenshot displays the GitHub profile for **neptune.ml**. The profile header includes the repository count (18), follower count (1), and project count (0). Below this, a section titled "Pinned repositories" lists six repositories in a grid. Each repository card shows the repository name, a brief description, the programming language (Python), and the number of stars and forks.

Repository Name	Description	Language	Stars	Forks
open-solution-salt-identification	Open solution to the TGS Salt Identification Challenge	Python	99	43
open-solution-home-credit	Open solution to the Home Credit Default Risk challenge	Python	264	104
steppy	Lightweight, Python library for fast and reproducible experimentation	Python	52	23
open-solution-mapping-challenge	Open solution to the Mapping Challenge	Python	54	15
open-solution-googleai-object-detection	Open solution to the Google AI Object Detection Challenge	Python	20	10
neptune-quick-start	Start models' training with neptune right now	Python	3	0

1. Data check – Outlier check

```
X['CODE_GENDER'].replace('XNA', np.nan, inplace=True)
X['DAYS_EMPLOYED'].replace(365243, np.nan, inplace=True)
X['NAME_FAMILY_STATUS'].replace('Unknown', np.nan, inplace=True)
X['ORGANIZATION_TYPE'].replace('XNA', np.nan, inplace=True)

bureau['AMT_CREDIT_SUM'].fillna(self.fill_value, inplace=True)
bureau['AMT_CREDIT_SUM_DEBT'].fillna(self.fill_value, inplace=True)
bureau['AMT_CREDIT_SUM_OVERDUE'].fillna(self.fill_value, inplace=True)
bureau['CNT_CREDIT_PROLONG'].fillna(self.fill_value, inplace=True)
```

2. Feature engineering – Ratio features

Ratio feature : A per B

```
X['annuity_income_percentage'] = X['AMT_ANNUITY'] / X['AMT_INCOME_TOTAL']
X['car_to_birth_ratio'] = X['OWN_CAR_AGE'] / X['DAYS_BIRTH']
X['car_to_employ_ratio'] = X['OWN_CAR_AGE'] / X['DAYS_EMPLOYED']
```

- 연금 / 전체 수입
- 차 소유한 나이 / 총 생애 일수
- 차 소유하 나이 / 총 근무 일수

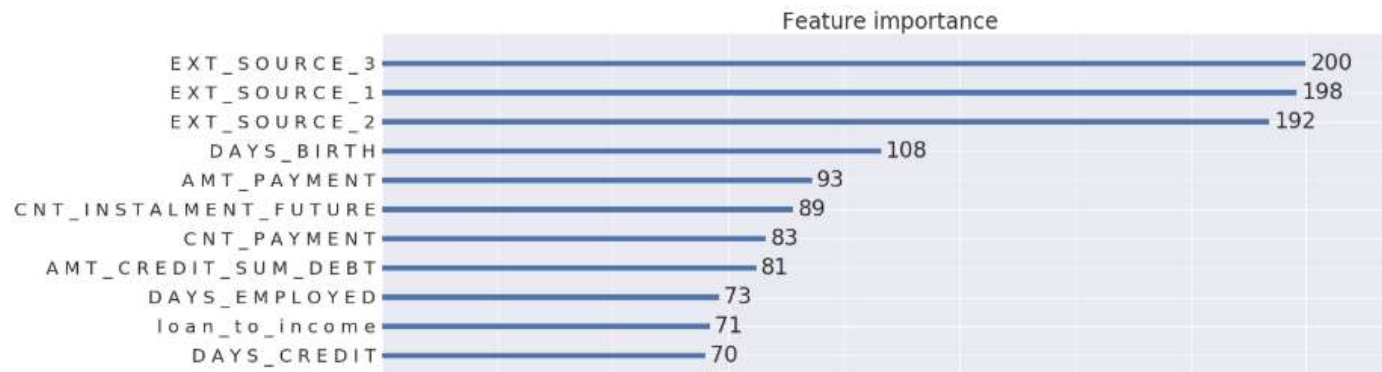
```
X['annuity_income_percentage'] = X['AMT_ANNUITY'] / X['AMT_INCOME_TOTAL']
X['car_to_birth_ratio'] = X['OWN_CAR_AGE'] / X['DAYS_BIRTH']
X['car_to_employ_ratio'] = X['OWN_CAR_AGE'] / X['DAYS_EMPLOYED']
X['children_ratio'] = X['CNT_CHILDREN'] / X['CNT_FAM_MEMBERS']
X['credit_to_annuity_ratio'] = X['AMT_CREDIT'] / X['AMT_ANNUITY']
X['credit_to_goods_ratio'] = X['AMT_CREDIT'] / X['AMT_GOODS_PRICE']
X['credit_to_income_ratio'] = X['AMT_CREDIT'] / X['AMT_INCOME_TOTAL']
X['days_employed_percentage'] = X['DAYS_EMPLOYED'] / X['DAYS_BIRTH']
X['income_credit_percentage'] = X['AMT_INCOME_TOTAL'] / X['AMT_CREDIT']
X['income_per_child'] = X['AMT_INCOME_TOTAL'] / (1 + X['CNT_CHILDREN'])
X['income_per_person'] = X['AMT_INCOME_TOTAL'] / X['CNT_FAM_MEMBERS']
X['payment_rate'] = X['AMT_ANNUITY'] / X['AMT_CREDIT']
X['phone_to_birth_ratio'] = X['DAYS_LAST_PHONE_CHANGE'] / X['DAYS_BIRTH']
X['phone_to_employ_ratio'] = X['DAYS_LAST_PHONE_CHANGE'] / X['DAYS_EMPLOYED']
X['external_sources_weighted'] = X.EXT_SOURCE_1 * 2 + X.EXT_SOURCE_2 * 3 + X.EXT_SOURCE_3 * 4
X['cnt_non_child'] = X['CNT_FAM_MEMBERS'] - X['CNT_CHILDREN']
X['child_to_non_child_ratio'] = X['CNT_CHILDREN'] / X['cnt_non_child']
X['income_per_non_child'] = X['AMT_INCOME_TOTAL'] / X['cnt_non_child']
X['credit_per_person'] = X['AMT_CREDIT'] / X['CNT_FAM_MEMBERS']
X['credit_per_child'] = X['AMT_CREDIT'] / (1 + X['CNT_CHILDREN'])
X['credit_per_non_child'] = X['AMT_CREDIT'] / X['cnt_non_child']
for function_name in ['min', 'max', 'sum', 'mean', 'nanmedian']:
    X['external_sources_{}'.format(function_name)] = eval('np.{}'.format(function_name))(
        X[['EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3']], axis=1)

X['short_employment'] = (X['DAYS_EMPLOYED'] < -2000).astype(int)
X['young_age'] = (X['DAYS_BIRTH'] < -14000).astype(int)
```

2. Feature engineering – Product features

Product feature : A x B

- Feature importance 를 봤을 때, 상위 feature 들 중 numerical feature 끼리 곱하여 추가 함.



```
df['EXT_SOURCE_1_x_EXT_SOURCE_2'] = df['EXT_SOURCE_1'] * df['EXT_SOURCE_2']  
df['EXT_SOURCE_1_x_EXT_SOURCE_3'] = df['EXT_SOURCE_1'] * df['EXT_SOURCE_3']  
df['EXT_SOURCE_2_x_EXT_SOURCE_3'] = df['EXT_SOURCE_2'] * df['EXT_SOURCE_3']  
df['AMT_PAYMENT_x_EXT_SOURCE_3'] = df['AMT_PAYMENT'] * df['EXT_SOURCE_3']
```

2. Feature engineering – Addition or Subtraction features

Addition feature : $A + B$

Subtraction feature : $A - B$

- 중요한 feature 끼리 더하거나 빼서 새로운 feature 생성 .

```
# External sources
X['external_sources_weighted'] = X.EXT_SOURCE_1 * 2 + X.EXT_SOURCE_2 * 3 + X.EXT_SOURCE_3 * 4
for function_name in ['min', 'max', 'sum', 'mean', 'nanmedian']:
    X['external_sources_{}'.format(function_name)] = eval('np.{}'.format(function_name))(
        X[['EXT_SOURCE_1', 'EXT_SOURCE_2', 'EXT_SOURCE_3']], axis=1)
```


2. Feature engineering – New categorical features

특정 기준을 만족하느냐,
만족하지 않느냐로
Binary category 를 만들 수 있음.

```
bureau['bureau_credit_active_binary'] = (bureau['CREDIT_ACTIVE'] != 'Closed').astype(int)  
bureau['bureau_credit_enddate_binary'] = (bureau['DAYS_CREDIT_ENDDATE'] > 0).astype(int)
```

2. Feature engineering – Aggregation features

Category 와 numerical feature 의 조합으로 생성하며,
Category 각 그룹당 mean, median, variance, standard
deviation 을 feature 로 사용

```
group_object = credit_card.groupby(by=['SK_ID_CURR'])['AMT_DRAWINGS_ATM_CURRENT'].agg('sum').reset_index()
```

말그대로 조합이기 때문에,
엄청나게 만들어 낼 수 있음.

$$C(n, r) = \frac{n!}{r! (n - r)!}$$

2. Feature engineering – Categorical features

One-hot
encoding

Category 개수 만큼 column 이 늘어난다.
너무 오래 걸린다.

Label
encoding

자칫 bias ordering 문제가 생길 수 있다.

Lightgbm
Built-in

데이터셋이 크니 학습이 빠른 Lightgbm 을
쓰건데, 마침 카테고리를 처리하는 내장
알고리즘이 있구나! 이거다

2. Feature engineering – Categorical features

▸ Categorical Encoding Methods

build passing coverage unknown circleci failing DOI 10.5281/zenodo.1157110

A set of scikit-learn-style transformers for encoding categorical variables into

Important Links

Documentation: <http://contrib.scikit-learn.org/categorical-encoding/>

- Encoding 방법은 정말 많습니다.

- Backward Difference contrast
- BaseN
- Binary
- Hashing
- Helmert Contrast
- James-Stein Estimator
- LeaveOneOut
- M-estimator
- Ordinal
- One-hot
- Polynomial Contrast
- Sum Contrast
- Target encoding
- Weight of Evidence

2. Feature engineering – Fill missing values and infinite values

Numerical features

Lightgbm 은 missing value 를 빼고 tree split 을 한 다음, missing value 를 각 side 에 넣어봐서 loss 가 줄어드는 쪽으로 missing value 를 분류

- 0 로 채우지 말고, 내장 알고리즘 쓰기로 결정.
- (+) Infinite value 는 $1.2 * \text{max value}$
- (-) Infinite value 는 $1.2 * \text{min value}$

Categorical features

‘NAN’ 이라는 새로운 category 를 만들어서 채움

3. Feature selection – Use various approaches

이리저리 만들다 보니 약 2300개 features 생성됨.
이걸 다 쓸 것인가?

Feature
importance

Recursive
feature
elimination

Shap

4. Model development – Use LGBM

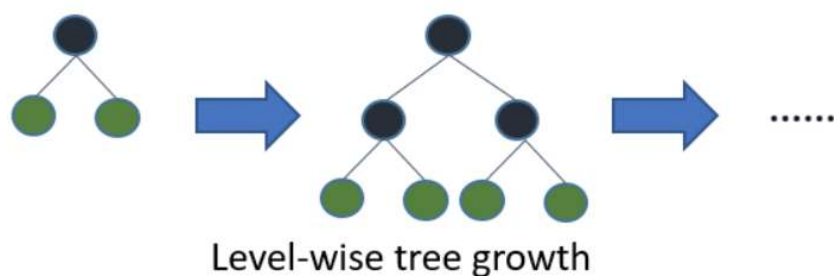
Why lightgbm?

- Faster training speed and higher efficiency
- Lower memory usage
- Better accuracy
- Parallel and GPU learning supported
- Capable of handling large-scale data

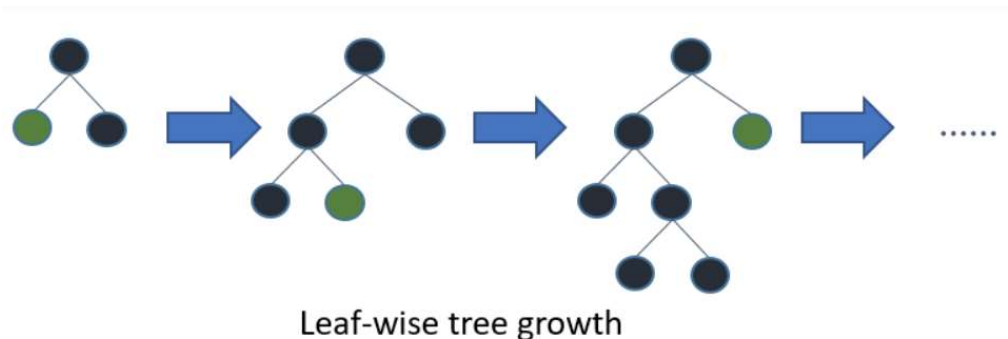
4. Model development – Use LGBM

What is different?

Most decision tree algorithm

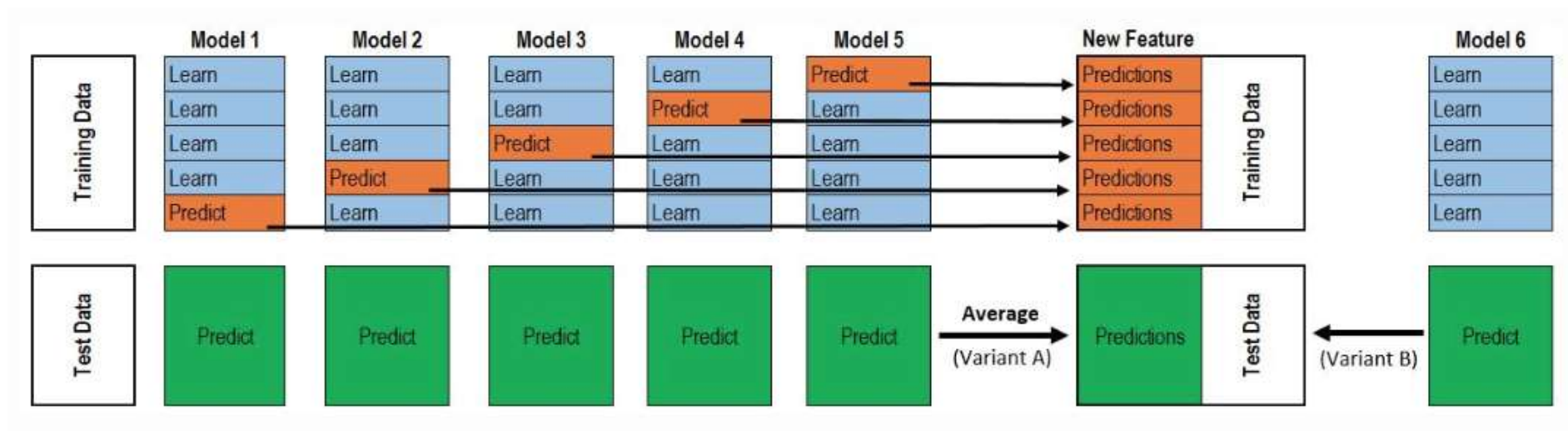


Lightgbm



종고 빠르다 쓰자

5. Training strategy – Ensemble is always answer

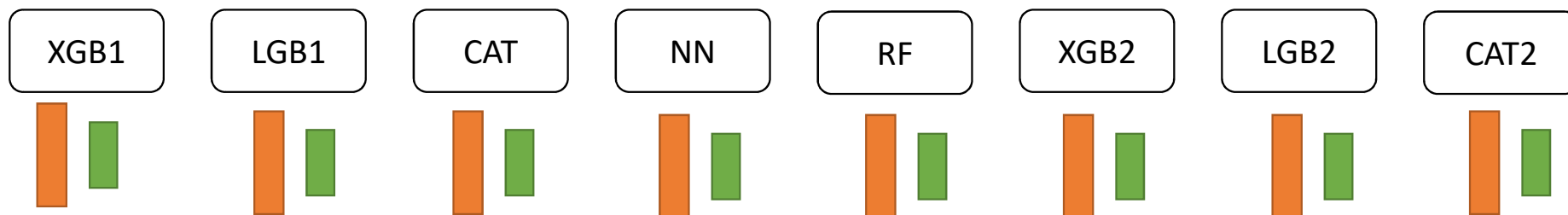
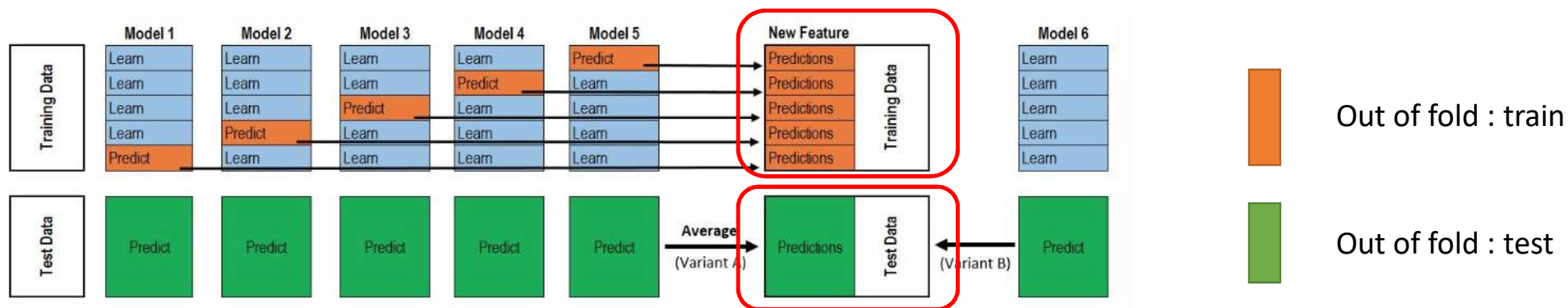


Sum of Weak learner is stronger than One strong learner.

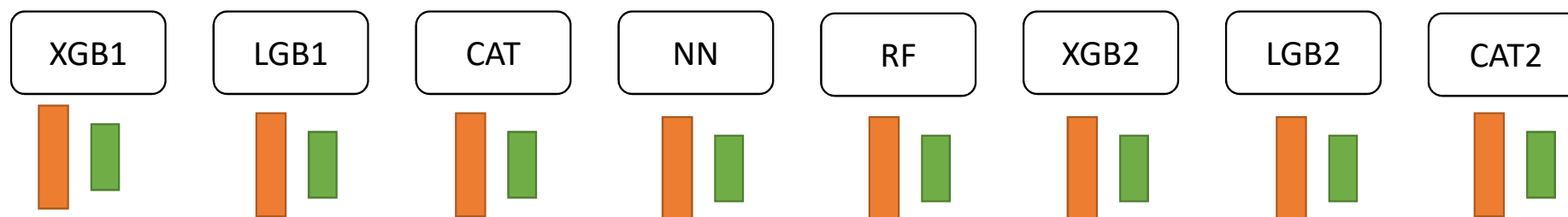
이렇게 해서 제출한
성적은?

동메달은 무슨...
그래도 TOP 15%
정도는 갔어요

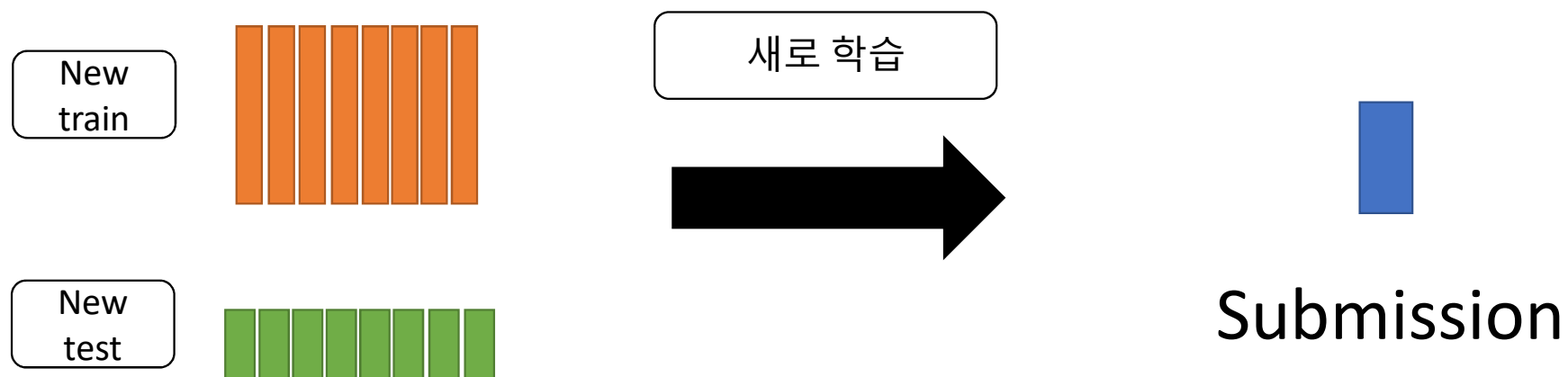
6. Stacking and Averaging



6. Stacking and Averaging

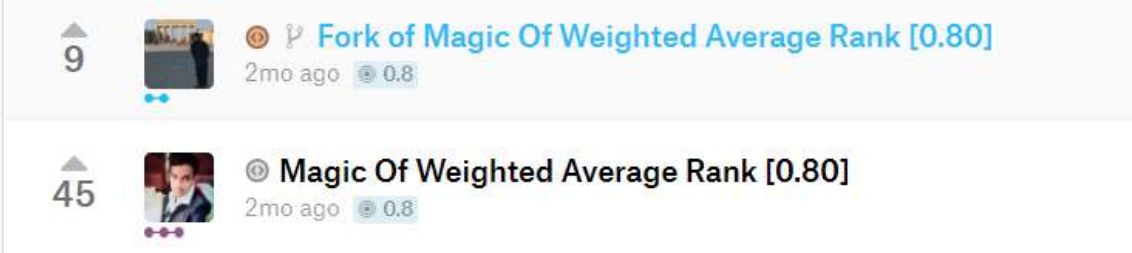
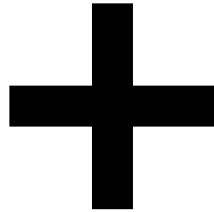


애네들을 feature 로 사용함



6. Stacking and Averaging


우리
Submission



Rank	Submission Name	Score	Time Ago
9	Fork of Magic Of Weighted Average Rank [0.80]	0.8	2mo ago
45	Magic Of Weighted Average Rank [0.80]	0.8	2mo ago

다른 캐글러
Submissions

Simple average or weighted average

이렇게 해서 제출한
성적은?

동메달권
들어갔습니다.

자, 이제 더 성능을
올리려면 뭘
해야할까요?

Feature
generation

Parameter
tuning

More stacking

자, 이제 더 성능을
올리려면 뭘
해야할까요?

Feature
generation

Parameter
tuning

More stacking

다 해야합니다 ^^

저는
Parameter tuning 만
선택했습니다.

이것이 동메달로 끝내게 한
main reason 입니다.

7. Hyper parameter tuning

For Faster Speed

- Use bagging by setting `bagging_fraction` and `bagging_freq`
- Use feature sub-sampling by setting `feature_fraction`
- Use small `max_bin`
- Use `save_binary` to speed up data loading in future learning
- Use parallel learning, refer to [Parallel Learning Guide](#)

For Better Accuracy

- Use large `max_bin` (may be slower)
- Use small `learning_rate` with large `num_iterations`
- Use large `num_leaves` (may cause over-fitting)
- Use bigger training data
- Try `dart`

Deal with Over-fitting

- Use small `max_bin`
- Use small `num_leaves`
- Use `min_data_in_leaf` and `min_sum_hessian_in_leaf`
- Use bagging by set `bagging_fraction` and `bagging_freq`
- Use feature sub-sampling by set `feature_fraction`
- Use bigger training data
- Try `lambda_l1`, `lambda_l2` and `min_gain_to_split` for regularization
- Try `max_depth` to avoid growing deep tree

7. Hyper parameter tuning

Grid Search

- Parameters Grid space 를 만들어서 성능 확인
- 규칙적으로 optimum 을 찾아갈 수 있다.
- 하지만 grid size 에 따라 combination 이 너무 많아진다. Computational cost!
- 내가 어떤 영역을 잡느냐에 따라 optimum을 제대로 못 찾을 수 있다.

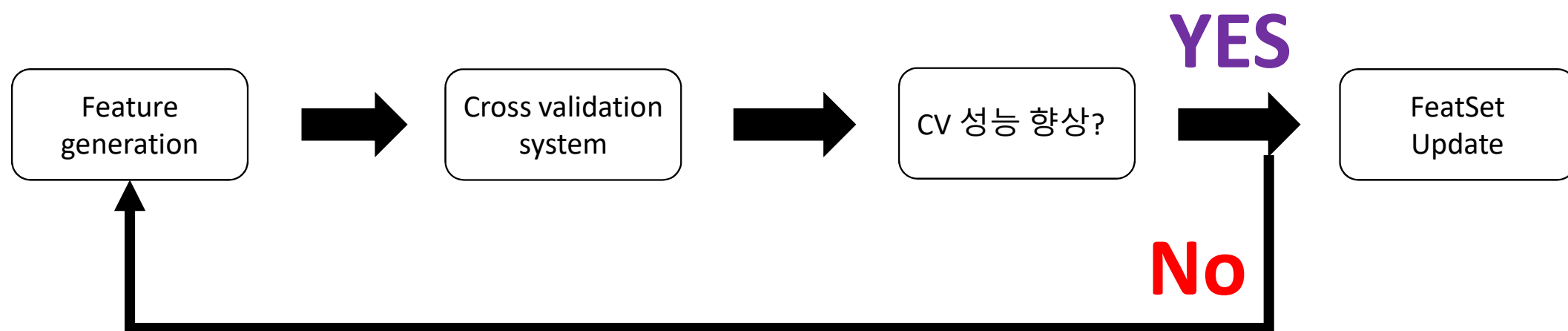
Randomized Search

- 각 Parameter 들의 range 를 만들어 그 안에서 임의로 숫자를 뽑아서 성능 확인
- 규칙적이지 않다. 하지만 trial 의 수는 많이 줄어 들 수 있다.
- 마찬가지로 내가 잡아주는 range 에 따라 optimum 을 제대로 못 찾을 수 있다.

Bayesian Optimization

- 성능 = $F(\text{parameters})$ 라는 함수를 가정하여, 그 함수의 형태를 추정하면서 global optimization 을 찾아가는 것.
- Bayesian 의 prior 개념이 들어감.
- 여러 trial 을 시도하면서, prior 를 이용해 global optimum 을 찾아가는 형태.
- Python package 가 있어서 사용하기 쉬움.
- 여러 trial 을 하면서 찾아가므로 기본 computational cost 가 있음.

Lesson1 - Feature generation 과 Cross-validation 은 함께 해야 한다.

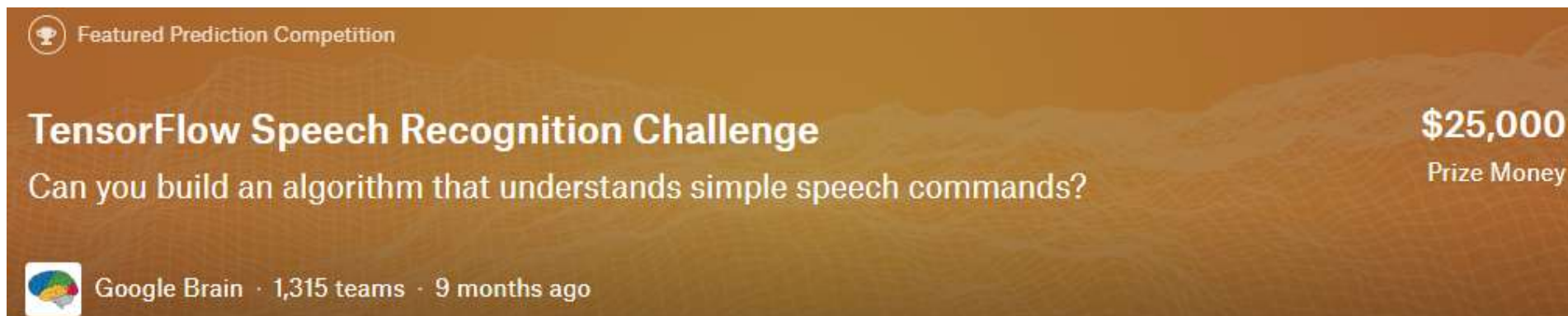


Bottom-Up Feature selection

Lesson2 - 모든 결과를 기록해야 한다.

날짜	Featset version	Feature 설명	성능(ROC)	Choose?
2018/09/09	FeatSet 1	Initial features	0.660	
2018/09/11	FeatSet 2	FeatSet1 + Time features	0.680	O
2018/09/14	FeatSet 3	FeatSet2 + Ratio features	0.700	O
2018/09/15	FeatSet 4	FeatSet3 + New A category	0.690	X
2018/09/16	FeatSet 5	FeatSet3 + New B category	0.720	O
2018/09/17	FeatSet 6	FeatSet5 + other encoding	0.760	O
....

딥러닝 컴퍼티션들 – Image recognition and classification




Featured Prediction Competition

TensorFlow Speech Recognition Challenge

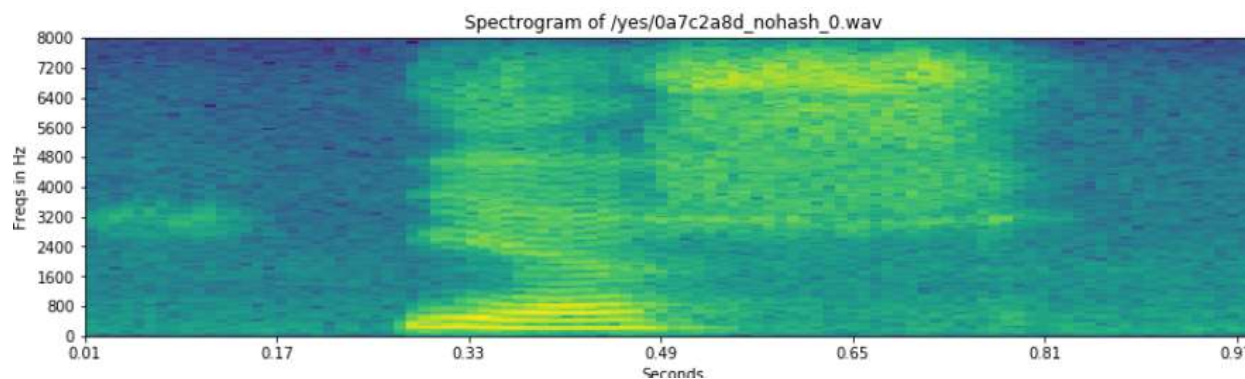
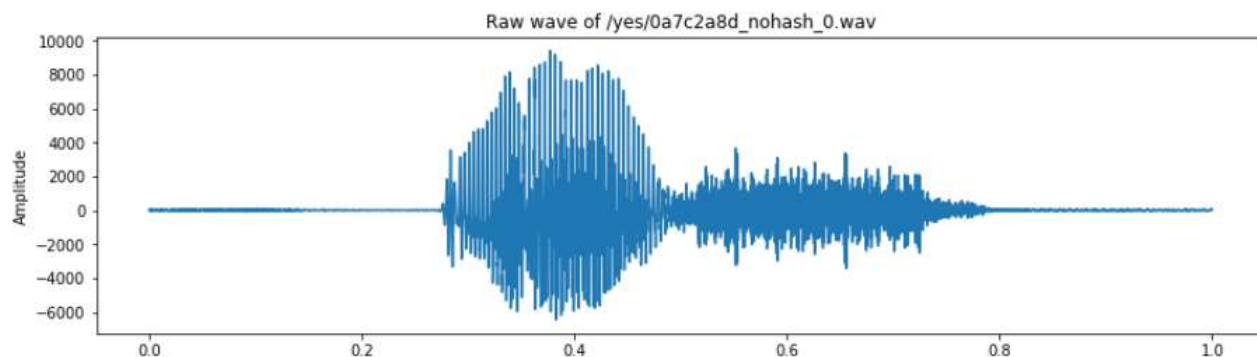
Can you build an algorithm that understands simple speech commands?

\$25,000
Prize Money

 Google Brain · 1,315 teams · 9 months ago

Yes, no, up, down, left, right, on, off, stop, go, silence, others
위 단어들을 구분할 수 있는 machine 을 만들어라!

Solution process (1) 더 많은 feature 을 위한 1D -> 2D 변환



Solution process (2) 2D-CNN 을 사용한 이미지 학습

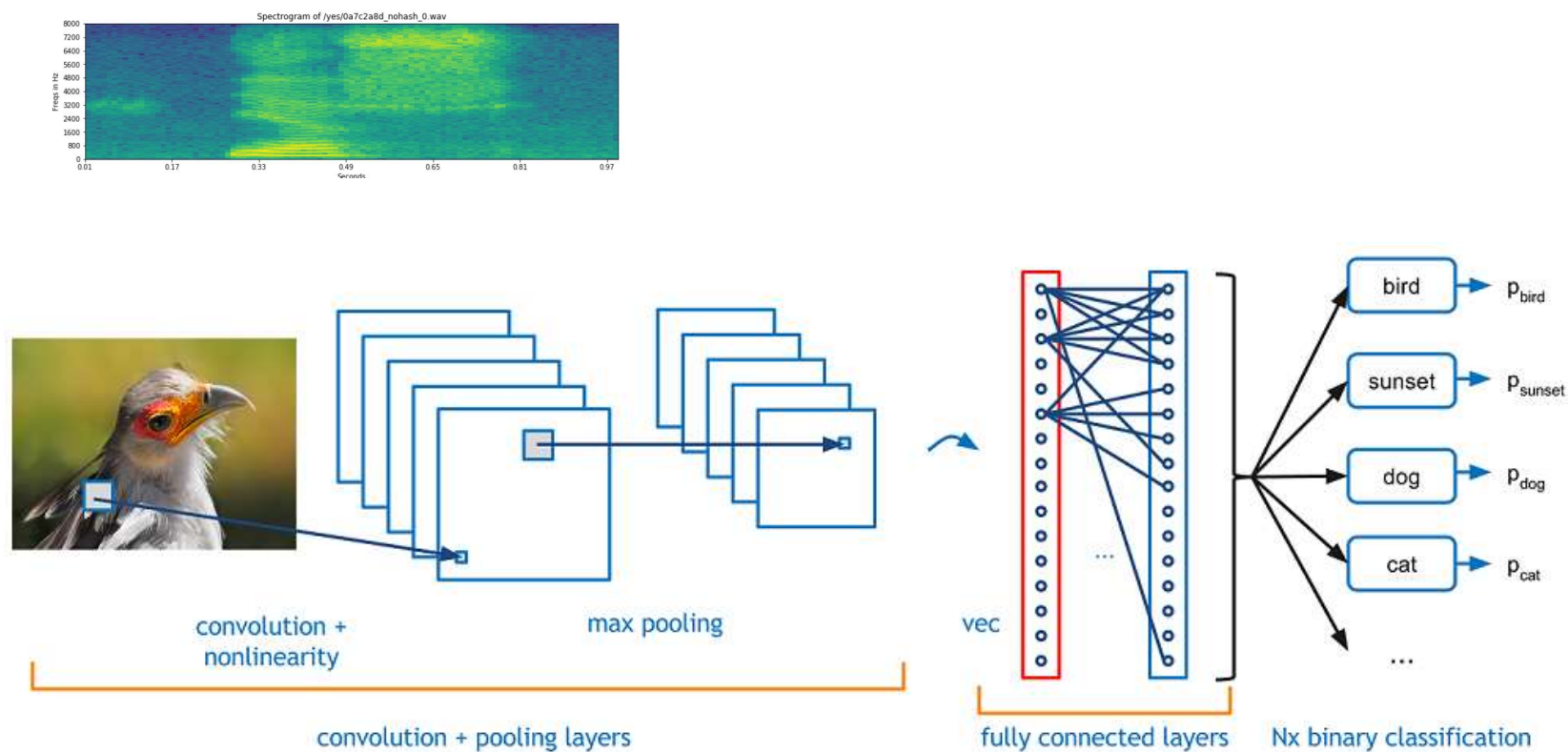


Image feature extraction

Integration of features

2D-CNN은 어떻게 만드나?

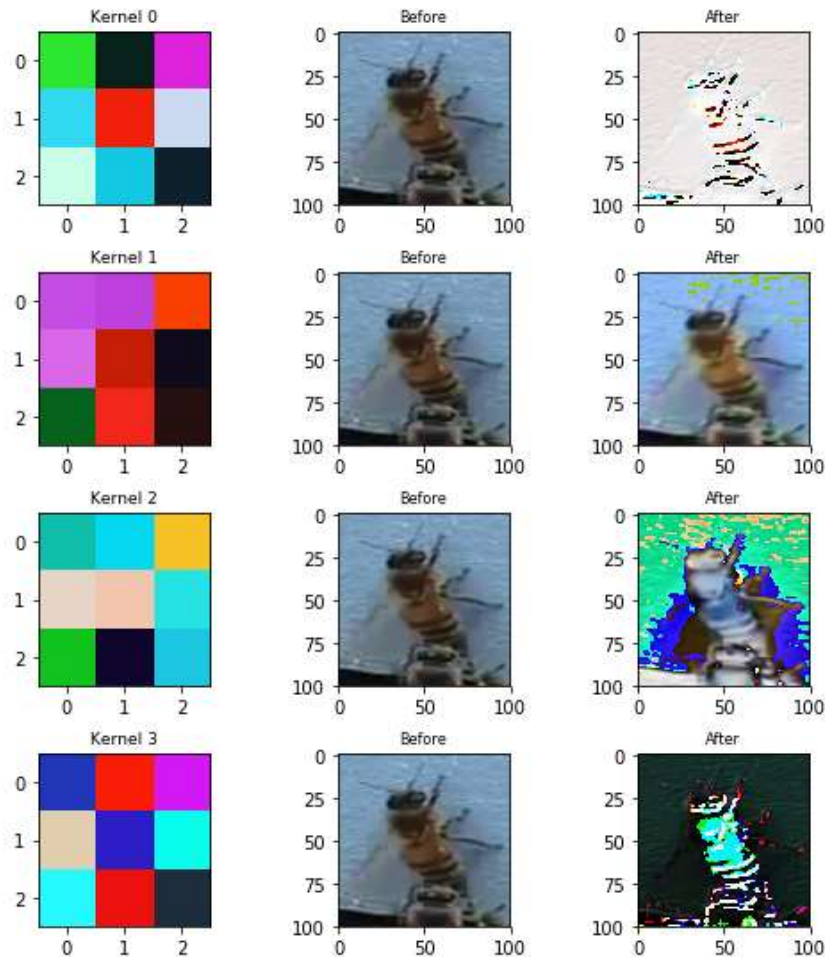


+



Keras

최신 동향 – Pre-trained model 사용

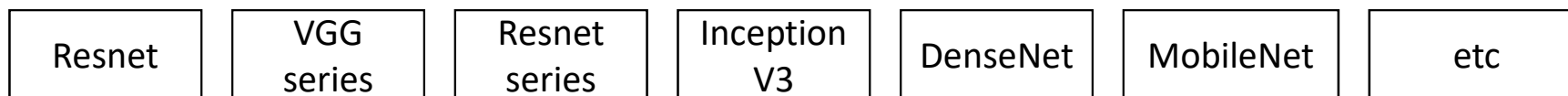
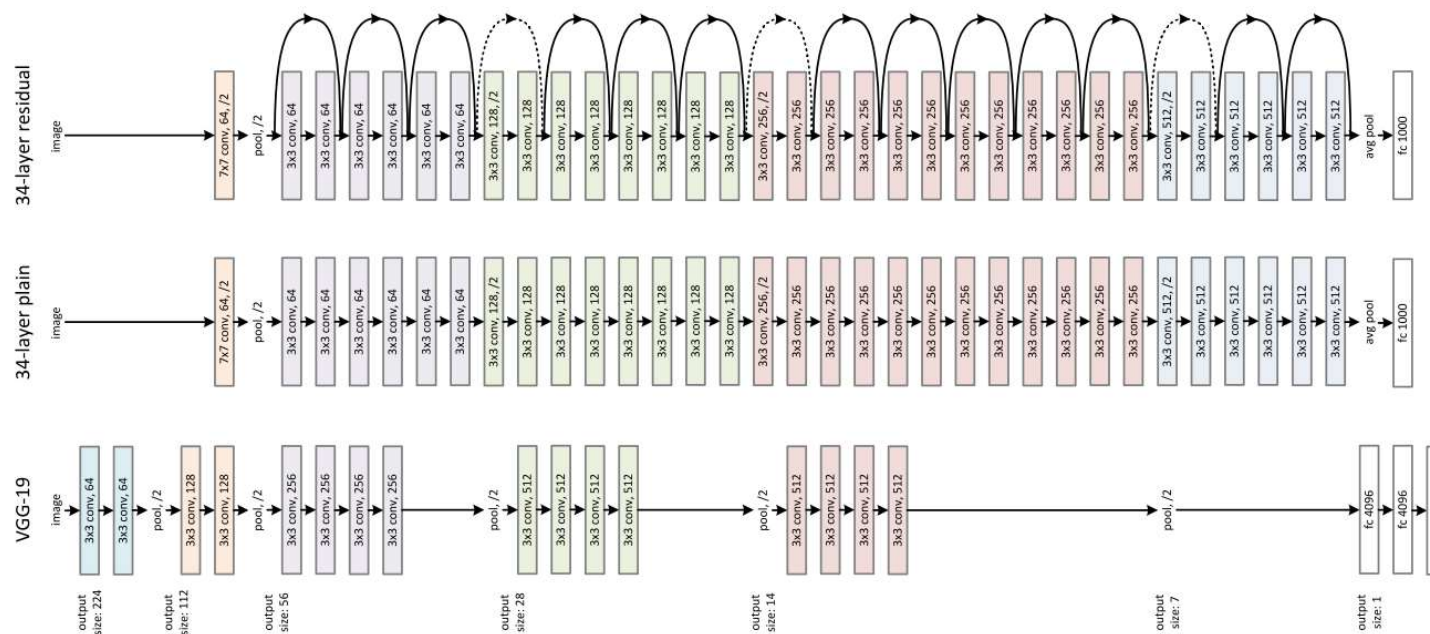


CNN 은 Image 에서 중요한 feature
를 뽑아내는 하나의 도구

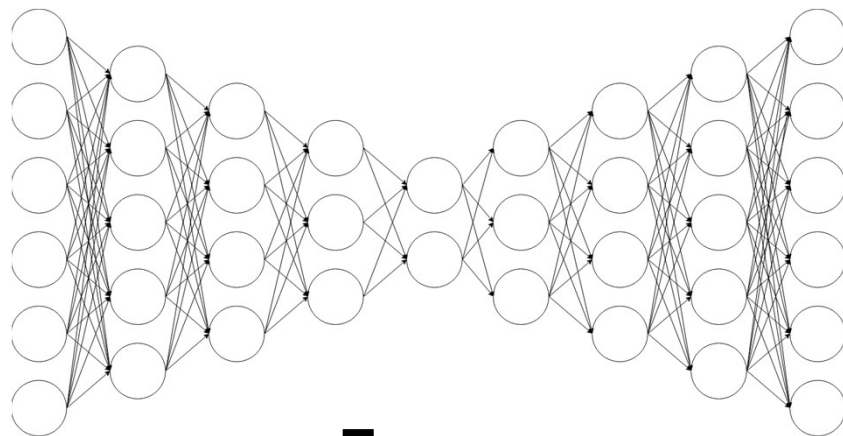


Image 에서 feature 를 잘
뽑아내게 학습된 model 을
가져와 쓰자

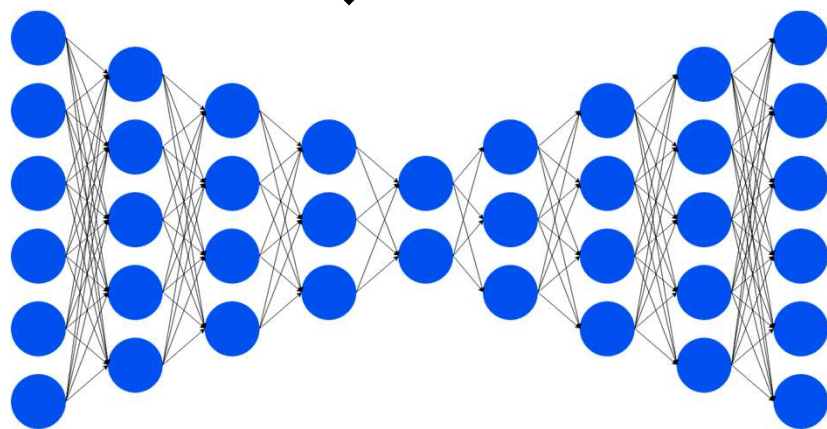
최신 동향 – Pre-trained model 사용



최신 동향 – Pre-trained model 사용



Training



Non-trained
-> random numbered(initialized)
weights

Trained
-> Optimized weights

사용방법? – Use built-in function

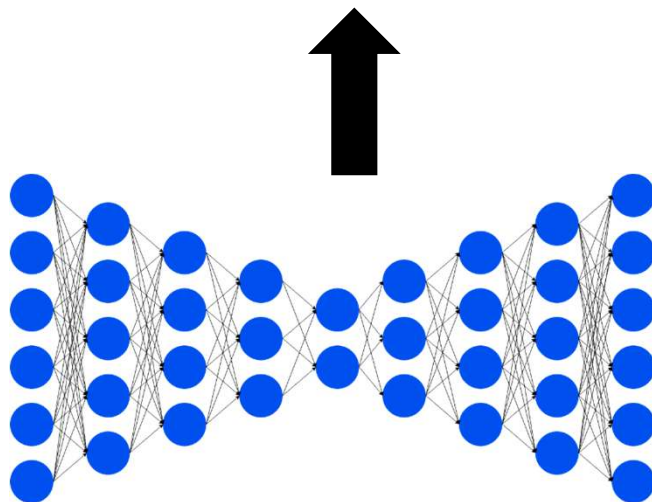
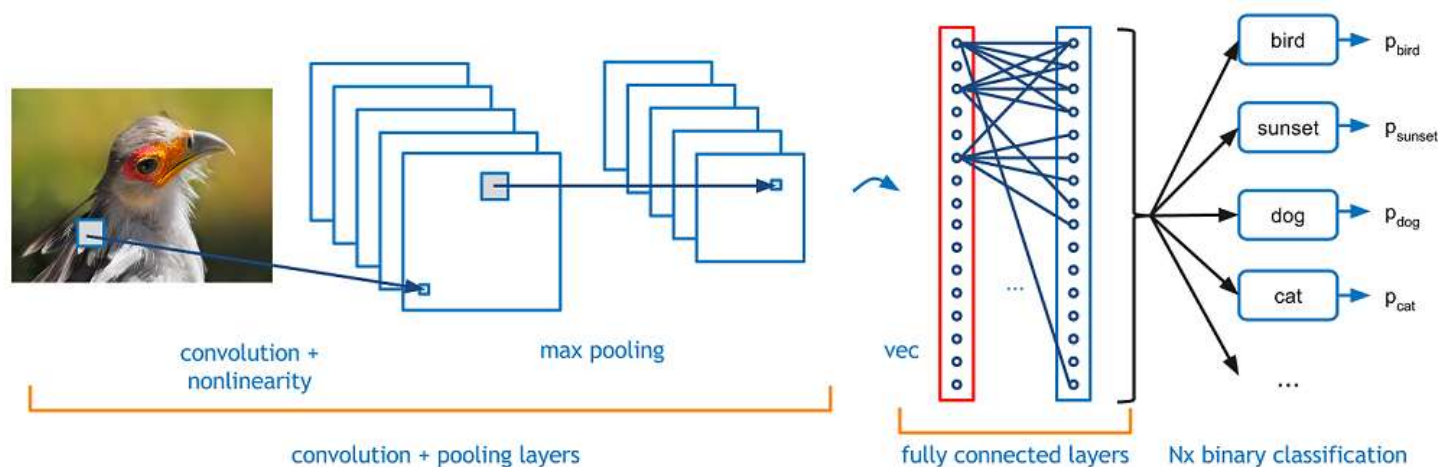
```
from keras.applications.resnet50 import ResNet50
from keras.preprocessing import image
from keras.applications.resnet50 import preprocess_input, decode_predictions
import numpy as np

model = ResNet50(weights='imagenet')

img_path = 'elephant.jpg'
img = image.load_img(img_path, target_size=(224, 224))
x = image.img_to_array(img)
x = np.expand_dims(x, axis=0)
x = preprocess_input(x)

preds = model.predict(x)
# decode the results into a list of tuples (class, description, probability)
# (one such list for each sample in the batch)
print('Predicted:', decode_predictions(preds, top=3)[0])
# Predicted: [(u'n02504013', u'Indian_elephant', 0.82658225), (u'n01871265', u'tusker', 0.1122357), (u'n02504458
```

사용방법? – Use built-in function



Yes, no, up, down, left,
right, on, off, stop, go,
silence, others

이런 지식을 캐글에서
지식을 얻으려면?

캐글의 위대한 유산 – 공유 정신

write-ups

1st Bojan Tunguz: [I am speechless](#)

1st Bojan Tunguz: [1st Place Solution](#)

2nd Maxwell: [2nd place solution \(team ikiri_DS \) \[Github\]](#)

2nd Giba: [Congratulations, Thanks and Finding!!!](#)

3rd alijs: [3rd place solution](#)

4th Shubin: [4th place sharing and tips about having a good teamwork experience](#)

5th narsil: [Overview of the 5th solution \[+0.004 to CV/Private LB of your model\]](#)

7th Abdelwahed Assklou: [7th solution - **Not great things but small things in a great way.**](#)

8th Xuan Cao: [8th Solution Overview](#)

9th MichaelP: [#9 Solution](#)

10th nlgn: [10th place writeup](#)

12th zr: [#12 solution](#)

13th Μαριος Μιχαηλιδης KazAnova: [13th place - time series features](#)

14th seagull: [#14 solution](#)

16th propower: [The 16th Solution](#)

17th Qinghui Ge: [17th place mini writeup](#)

19th AllMight: [# 19th solution](#)

24th Arthur Llau: [24th place - Simple Solution with 7 Models.](#)

27th nyanp: [Pseudo Data Augmentation \(27th place short writeup\) \[Github\]](#)

32th arnowaczynski: [Story behind the 32th place \(top 1%\) with 2 submissions](#)

48th James Davis: [Simple feature that made public kernels top 50 \(and thanks!\)](#)

Register with just one click:

We won't share anything without your permission

Google

Facebook

Yahoo

Manually create an account:

Email

Password

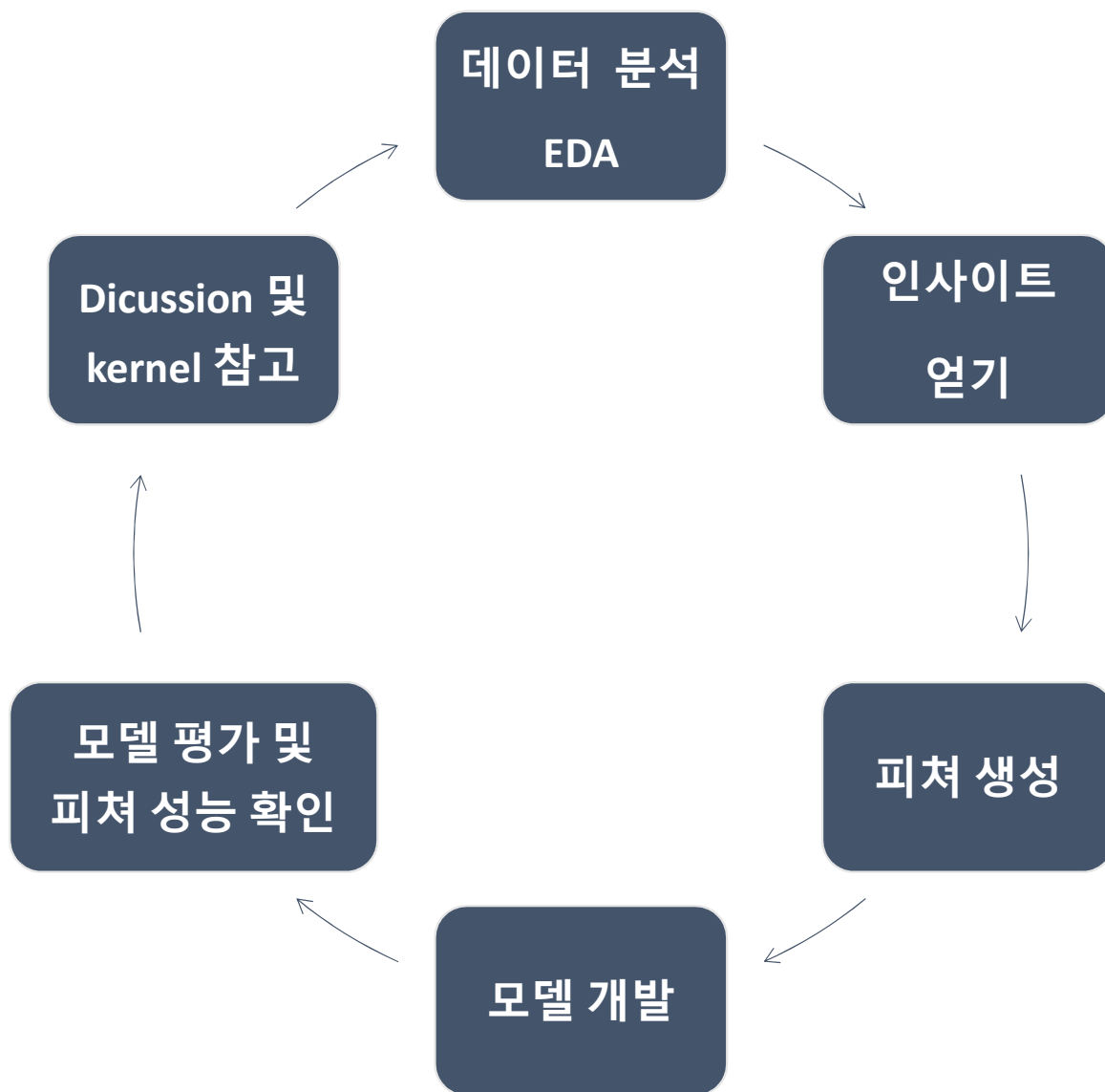
Register

구글,
페이스북,
야후 아이디로
가입만 하시면
됩니다

몇가지 천기 누설!!!!

- 투자한 시간은 배신하지 않는다.(from 김현우)
 - Keep going!
- 데이터 탐색 >>> 파라미터 튜닝 (from 김연민)
- Garbage in, Garbage out
 - Show me the feature!
- See kernel, See discussion
- Many weak learner win one strong learner
 - ensemble
- Make Cross-validation system along with LB

캐글에서
메달 따는 왕도는??



×

100,000

캐글 코리아

Kaggle Korea

Non-Profit Facebook Group Community

함께 공부해서, 함께 나눕시다
Study Together, Share Together

들어주셔서
감사합니다!