

자료분석특론2
데이터 분석 보고서

212STG18 예지혜

목차

A. Levels Fyi Salary data

- I. 자료 설명**
- II. EDA 및 전처리**
- III. 모형 탐색 과정**
- IV. 최종 모형**

B. Mobilephone Image data

- I. 자료 설명**
- II. EDA 및 전처리**
- III. 모형 탐색 과정**
- IV. 최종 모형**

A. Levels Fyi Salary data

I. 자료 설명

i. 자료 설명

데이터 과학 분야와 STEM(Science, Technology, Engineering, and Mathematics) 분야의 연봉에 대한 데이터로, Levels.fyi에서 스크래핑하여 캐글에 올려놓은 자료이다. Levels.fyi는 IT 업계의 각 회사별 직책과 연봉을 비교해주는 미국 스타트업으로 IT 종사자들의 더 나은 커리어 의사결정을 돕는다.

반응 변수로는 회사로부터 받는 보상에 대해 연봉, 스톡 그랜트, 보너스가 있고, 이를 모두 합친 total yearly compensation 변수가 있다. 설명 변수로는 회사 이름, 회사 내 직책, 직무, 지역, 경력, 회사 내 경력, 학력, 인종 등이 있다. 총 62642개의 관측치와 29개의 변수가 있다.

ii. 자료 출처

- 원자료 출처 : Levels.fyi

- 스크래핑된 데이터 :

https://www.kaggle.com/jackogozaly/data-science-and-stem-salaries?select=Levels_Fyi_Salary_Data.csv

iii. 분석 목표

관측치가 가장 많고 유명한 아마존, 마이크로소프트, 구글, 페이스북, 애플 다섯개의 회사에 다니는 사람들의 직책, 직무, 경력, 학력 등 관측치에 대한 정보를 통해 회사로부터 받는 연봉, 보너스 등을 예측해보고 중요한 요인을 탐색한다.

II. EDA 및 전처리

i. 반응변수 : totalyearlycompensation, basesalary, stockgrantvalue, bonus

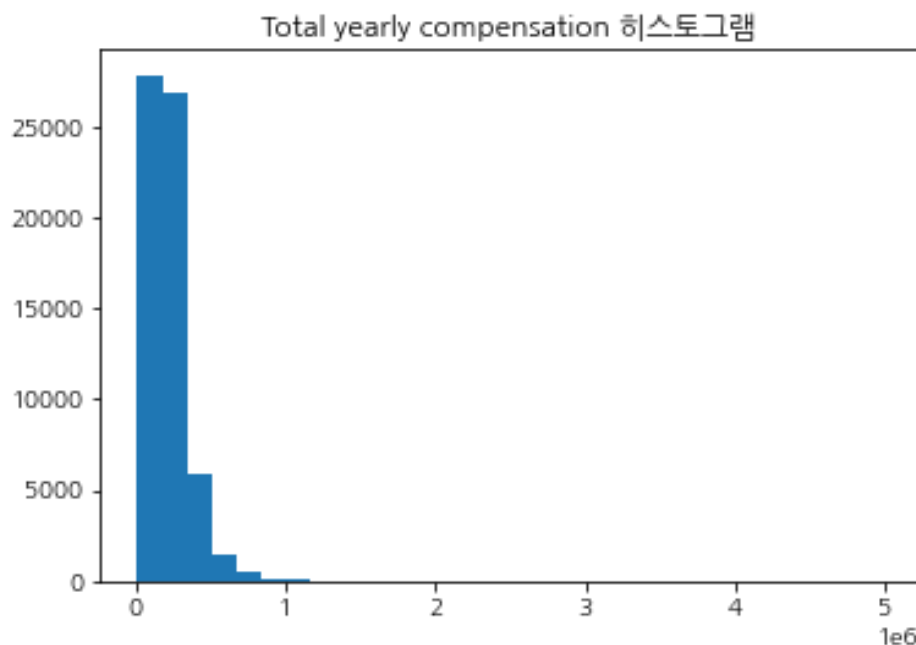
먼저 어떤 변수를 반응변수로 이용할지 결정하기 위해 자료를 탐색하였다.

- 요약 통계량

	totalyearlycompensation	basesalary	stockgrantvalue	bonus
Mean	216,300	136,687	51,486	1,9335
Std	138,034	61,369	81,874	26,781
Min	10,000	0	0	0
Median	188,000	140,000	25,000	14,000
Max	4,980,000	1,659,870	2,800,000	1,000,000

(소수점 첫째자리에서 반올림)

연봉이 가장 직관적이고 일반적으로 외부 영향이 적으나, 0이라는 이상치가 있다. 스톡그랜트와 보너스도 마찬가지이며, 이상치의 영향을 줄이고, 전반적인 보상을 평가하고자 total yearly compensation을 반응변수로 사용하겠다. 이 변수는 나머지 세 변수의 합산으로 추정되나, 확인해 본 결과 20%의 관측치에서 불일치하는 것으로 나타났다. 이는 나머지 세 변수의 이상치의 영향으로 보이므로 total yearly compensation 변수에는 이상이 없는 것으로 간주하고 분석을 진행하였다. 앞으로의 분석에서는 '연간 전체 보상' 이라고 칭하겠다.

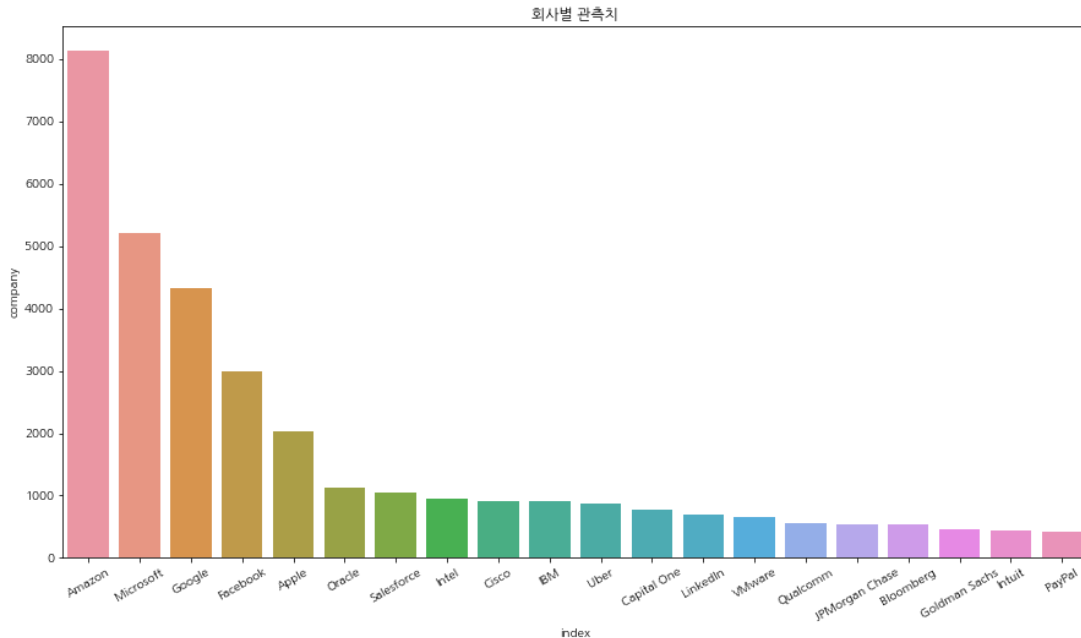


연간 전체 보상의 히스토그램을 보면, 왼쪽으로 매우 치우친 상태이다. 분산이 매우 크고 오른쪽 꼬리가 길기 때문에 분석에 있어서 로그 변환도 고려할 것이다.

ii. 설명변수

1) 회사 (company)

회사 개수는 총 1631개로, 매우 많으나 실제로 관측치가 충분한 회사는 많지 않았다.



아마존이 압도적으로 많은 관측치를 가지고 있으며, top 5 회사 뒤로는 관측치가 상대적으로 적기 때문에 Amazon, Microsoft, Google, Facebook, Apple 총 다섯 개 회사에 대해서만 분석하기로 결정하였다.

2) 직급 (level)

직급 또한 2923개로 종류가 매우 많은데, 이는 회사별로, 직무별로 다른 직급 체계를 가지기 때문이다. 이후, 직무에 대해 살펴보면 소프트웨어 엔지니어 관측치가 대부분이기 때문에 직급 또한 그와 관련된 관측치가 압도적으로 많다.

Top 5 회사에 대해서만 분석할 것이므로 해당 데이터에 대해 살펴보면, 직급은 325개의 종류가 있고 결측치가 1개 존재한다. 직급은 회사 내에서의 순서이기 때문에 보상에 직접적인 영향을 미치므로 해당 직급의 위치에 대한 정보를 부여할 필요가 있다. 직무에 따라라도 직급이 다양하지만 해당 데이터는 소프트웨어 엔지니어에 매우 치우쳐있으므로 해당 직무에 대한 직급 정보를 파악해보았다. 아래의 자료는 levels.fyi에서 제공하는 회사별 소프트웨어 엔지니어의 직급 순서이다. 회사간 비교는 모은 자료를 통해 levels.fyi가 매칭한 것이다. 이 정보에서 다른 명칭의 동일 직무를 통일시켰고, 마이크로소프트의 경우 다른 회사와 다르게 숫자가 60대 근처이므로 이를 맞춰주었다.

× Apple	× Amazon	× Google	× Facebook	× Microsoft
ICT2 Junior Software Engineer	SDE I L4	L3 SWE II	E3	SDE 59
ICT3 Software Engineer	SDE II L5	L4 SWE III	E4	SDE II 60
ICT4 Senior Software Engineer	SDE III Senior SDE L6	L5 Senior SWE	E5	Senior SDE 61
ICT5	Principal SDE L7	L6 Staff SWE	E6	62
ICT6	Senior Principal SDE L8	L7 Senior Staff SWE	E7	Senior SDE 63
Distinguished Engineer	Distinguished Engineer L10	L8 Principal Engineer	E8	Principal SDE 64
Senior Distinguished Engineer		L9 Distinguished Engineer	E9	65
Engineering Fellow		L10 Google Fellow		66
				67
				Partner 68
				69
				70
				Distinguished Engineer 80
				Technical Fellow

[levels.fyi의 소프트웨어 엔지니어 직급 정보]

실제 데이터를 살펴보면 정제되지 않은 형태의 직급 정보가 많았기 때문에, 숫자 정보를 최대한 활용하였다. 먼저, 숫자로 포함하지 않은 형태 중 위의 정보에 해당하는 경우, 숫자를 포함하는 동일 직무로 바꿔주었다. 그 다음 숫자 정보만을 추출하였고, 숫자가 매우 큰 경우 마이크로소프트로 간주하여 다른 회사와 비슷한 레벨로 맞춰주었다.

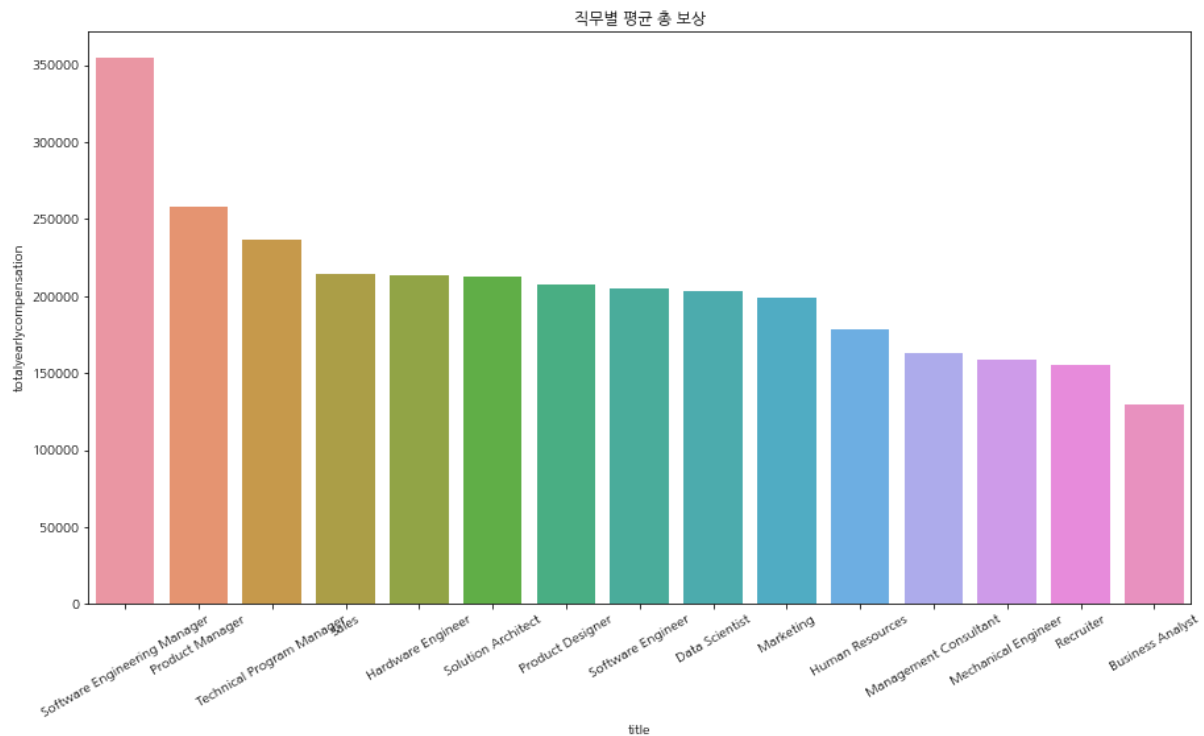
회사를 매칭하여 위의 정보를 활용하면 좋았겠지만, 다른 직무에 대한 직급도 다양하고, 위의 정보를 완벽히 반영하기 어려워 숫자 정보만 활용하였으며, 완벽하지 않은 전처리라 해당 변수를 사용한 모델링과 사용하지 않은 모델링을 비교해보고, 사용한 모델링의 성능이 훨씬 좋은 것을 확인하여 숫자 정보라도 활용하기로 결정하였다.

위의 정보에 해당하지 않고, 숫자 정보도 없는 경우는 하나하나 전처리하기 어려워 가장 많은 레벨로 대체하였다. 결국치 또한 가장 많은 레벨인 'L5'로 간주하였다.

3) 직무 (title)

직무는 총 15가지였으며 소프트웨어 엔지니어가 다른 직무들의 10배 이상에 해당하고 전체 데이터 중 65%를 차지한다. 직무별 평균 '연간 전체 보상'을 확인해보니, 직무마다 편차가 보였으며 소프트웨어 엔지니어가 가장 높았다. 하지만 이는, 해당 직무에 대한 데이터가 대부분이고, '연간 전체 보상' 히스토그램을 통해 오른쪽 꼬리가 매우 긴 것을 확인하였으므로 해당 직무라고 많이 번다고 보기는 어렵다. 다만, 직무별 보상의 편차가 있으므로 해당 변수를 사용하며, 순서를 부여

하는 정보는 아니기 때문에 원핫 인코딩을 사용하였다.



4) 성별 (gender)

전체 데이터 중 성별에 대한 정보에는 약 30%의 결측치가 존재한다. 매우 큰 비율을 차지하지만 성별 정보를 활용하지 않기에는 성별 간 임금 격차가 존재하기 때문에 결측치를 other 그룹에 넣어 분석하였다. 이상치 또한 결측치와 동일하게 처리하였다.

5) 인종 (race)

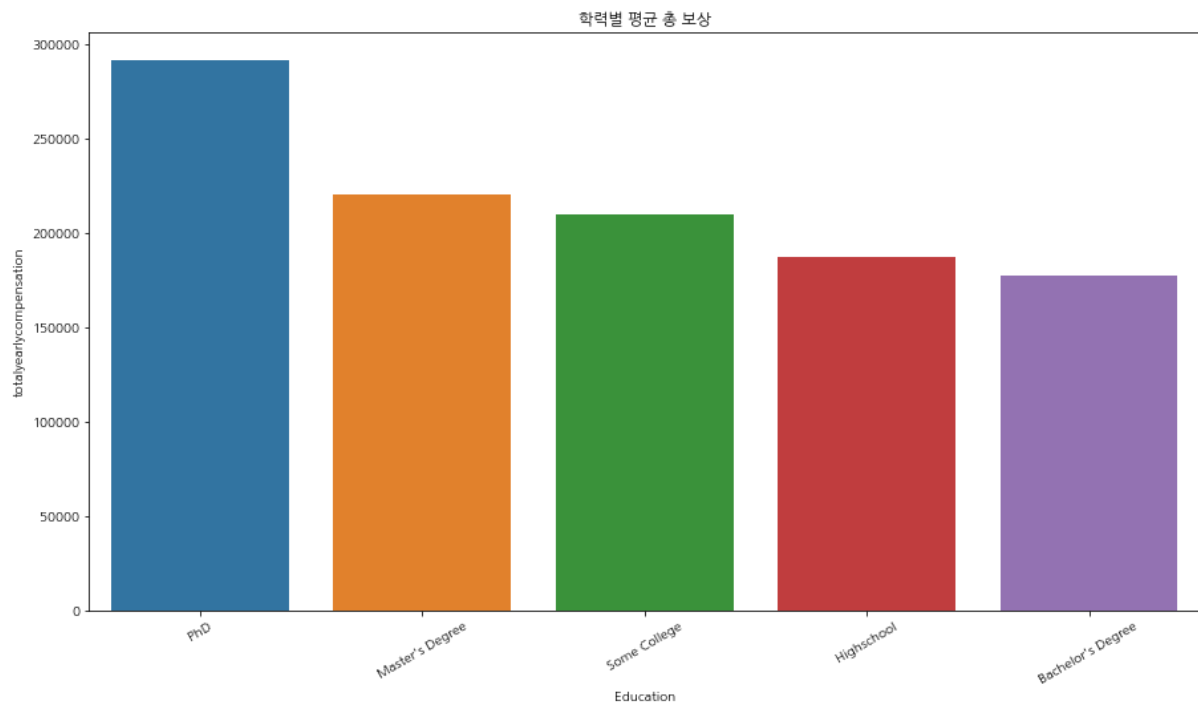
인종 정보는 Top5 데이터의 경우, 65%의 정보가 결측치로 없는 정보가 더 많았다. 원핫 인코딩된 변수인 Race_인종 변수를 활용해 채울 수 있는지 확인해보았으나 하나의 관측치를 빼고는 동일한 정보를 담고 있었다. 즉, race 변수에서 결측치인 관측치는 나머지 원핫 인코딩된 변수 중에도 1 인 변수가 없었다. 이 정보를 쓰지 않기에는 인종 간 격차가 존재하므로, 따로 결측치 처리를 하지 않고 원핫 인코딩된 변수만을 사용하였다.

6) 학력 (education)

학력 정보 또한 약 50%의 결측치가 존재하였고, 원핫 인코딩된 변수로 결측치를 채울 수 없었다. 학력 간 차이가 명확하므로 마찬가지로 원핫인코딩된 정보라도 사용하였다. 학력은

어찌보면 순서가 있을 수 있으나 미국에서의 college 의 영향이나, 고등학교 졸업이더라도 무조건 교육을 적게 받았다고 보기 어렵기 때문에 순서 인코딩과 원핫 인코딩 둘다 사용하여 비교해보았다. 그 결과 성능이 비슷하였고, 주관을 덜 개입하기 위해 원핫 인코딩을 사용하였다.

실제로 학력별 '연간 전체 보상'의 평균을 비교해보면 박사 졸업의 경우 가장 높지만, 그 이후는 고등학교 졸업이 학사 졸업보다 살짝 높기 때문에 순서 인코딩을 사용하지 않기로 결정하였다.



7) 지역 (location)

지역 정보도 1050 개의 범주가 있고, 관측치가 매우 적은 범주들이 많기 때문에, top 5 데이터 중 관측치가 1000 개 이상인 지역 8 개만 범주로 살렸다. 원핫 인코딩을 사용하여 모델에 반영하였다.

iii. 최종 데이터

분석 목표는 가장 유명하고 관측치가 많은 top5 회사에 다니는 사람들의 '연간 전체 보상'을 예측하는 것으로, 총 22,690 개의 관측치와 46 개의 설명변수이다. 원핫 인코딩된 변수를 제외하면 년도, 직급, 경력, 회사내 경력, 학력, 인증, 회사, 지역, 직무, 성별 총 10 가지 정보를 사용한다고 볼 수 있다.

III. 모형 탐색 과정

i. 학습 데이터와 테스트 데이터 분리

사이킷런 내장 함수를 이용하여 80%의 학습 데이터와 20%의 테스트 데이터로 분리하였다.

ii. 모형 탐색 과정

후보 모델로는 수업시간에 다룬 모든 회귀 모형을 다루었다. 선형 회귀, 릿지 회귀, 라쏘 회귀, 의사결정트리 회귀, 랜덤포레스트 회귀, 로지스틱 회귀, svm 회귀, svr 회귀, 그래디언트 부스팅 회귀 총 9 가지 모델이다. 디폴트 모델 적합 후, 성능이 괜찮은 모델 위주로 그리드 서치를 통해 파라미터를 최적화하였다. 성능은 RMSE 와 MAE 를 동시에 고려하였다.

앞서 전처리 과정에서 고민이 되었던 부분이 전처리한 직급 정보를 사용할 것인지, 학력을 순서 인코딩할 것인지 등이 있었다. 성능이 괜찮은 모델 위주로 직급 정보를 빼고 적합해보았을 때 성능이 매우 나빠졌고, 학력은 순서 인코딩이나 원핫 인코딩의 차이가 크지 않았기 때문에 원핫 인코딩을 사용하였다. 이 부분에 대한 모델은 성능 비교에 따로 넣지 않겠다.

타겟 변수인 '연간 전체 보상'은 오른쪽 꼬리가 매우 긴 형태였기 때문에 로그 변환도 고려하였다. 결과적으로 로그 변환이 효과적이었던 모델은 주로 선형 모형 기반의 모델들이었으며, 주로 성능이 좋은 트리 기반 모델에서는 크게 효과가 없었다. 이 부분은 표로 다시 정리하겠다.

언급된 모델들을 비교해 보았을 때 주로 트리 기반 모델의 성능이 좋아 최근에 개발된 캣부스트 모델도 시도하였다. 캣부스트 모델은 범주형 변수를 잘 처리하기 때문에 원핫 인코딩된 데이터와 라벨 인코딩된 데이터에 대해 범주형 변수라는 정보를 부여한 두 가지 방식의 모델을 적용하였다. 이에 대한 내용을 iii. 모형 성능 비교에서 정리해보겠다.

iii. 모형 성능 비교

1) 9 가지 후보 모델

모델 종류	파라미터	Test RMSE	Test MAE
선형 모델	-	101,100	62,688
릿지 회귀	(디폴트) $\alpha = 1$	101,100	62,681
	$\alpha = 10$	101,102	62,630
라쏘 회귀	(디폴트, 그리드서치) $\alpha = 0.1$	101,100	62,687
의사결정트리	-	93,731	56,005
랜덤포레스트	(디폴트) max_depth = None,	75,483	44,168

	min_samples_leaf = 1, n_estimators = 100		
	max_depth = 50, max_features = 8, min_samples_leaf = 2, n_estimators = 30	76,330	44,178
	max_depth = 50, max_features = 10, min_samples_leaf = 2	76,382	44,007
로지스틱 회귀	-	135,661	80,428
SVM	-	128,195	76,066
SVR	Kernel="poly", degree = 2, C=100, epsilon = 0.1, gamma="scale"	148,814	91,829
그라디언트 부스팅	max_depth=2, n_estimators=100, learning_rate=1.0	82,753	48,384
	max_depth=10, n_estimators=50, learning_rate=0.2	74,263	43,042
	max_depth=10, n_estimators=40, learning_rate=0.2	74,216	43,031

(소수점 첫째자리에서 반올림)

모델의 결과를 비교해보면, 주로 트리 계열 기반 모델인 랜덤포레스트와 그라디언트 부스팅 모델의 성능이 우수했다. 그라디언트 부스팅의 경우 그리드 서치를 진행할수록 성능이 조금씩 더 개선되었고, 랜덤포레스트는 오히려 test RMSE 가 증가하는 경향을 보였다. 파라미터를 확인해보면 학습 데이터에 과적합된 것으로 해석할 수 있다. 따라서 랜덤포레스트는 디폴트 모형을 최종 모형의 후보 모델로 선택하였다.

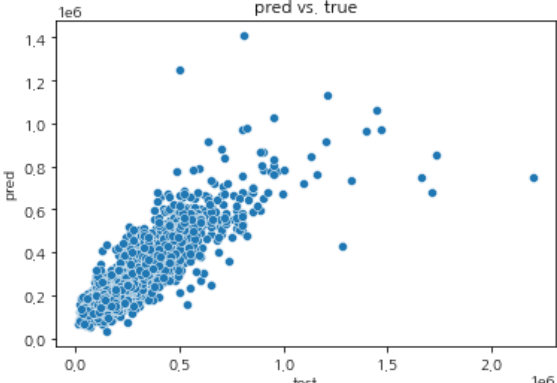
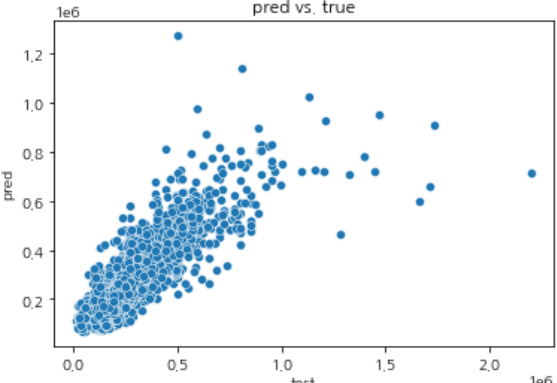
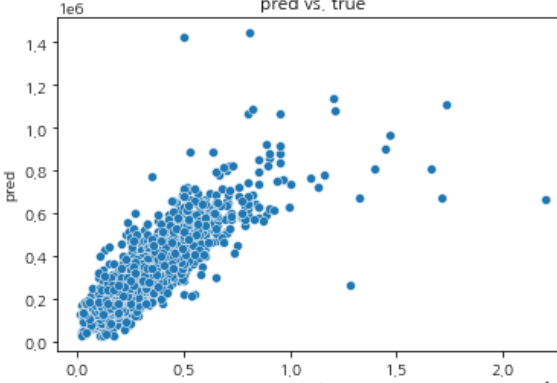
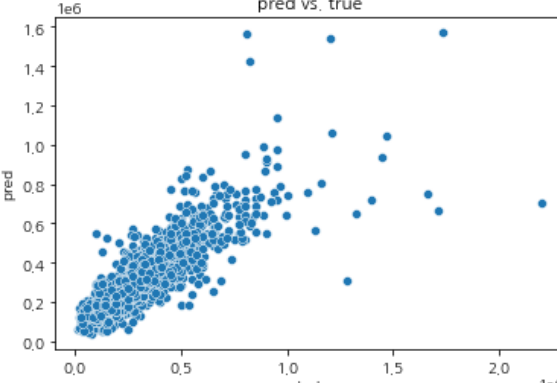
2) 로그 변환

모델 종류	파라미터	Test RMSE	Test MAE
선형 모델	-	98,835	57,164
릿지 회귀	(디폴트) $\alpha = 1$	98,841	57,163
라쏘 회귀	(디폴트) $\alpha = 0.1$	129,074	78,119
의사결정트리	-	93,463	56,530
랜덤포레스트	(디폴트) max_depth = None, min_samples_leaf = 1, n_estimators = 100	76,705	44,308
	max_depth = 20, max_features = 8, min_samples_leaf = 10, n_estimators = 30	85,149	46,947
SVM	-	106,841	63,584
SVR	Kernel="poly", degree = 2, C=100, epsilon = 0.1, gamma="scale"	118,274	68,727
그라디언트 부스팅	max_depth=2, n_estimators=100, learning_rate=1.0	94,604	45,777
	max_depth=5, n_estimators=50, learning_rate=0.5	77,689	43,598

(소수점 첫째자리에서 반올림)

로그 변환된 모델의 경우 성능이 원래부터 우수한 모델에는 별로 영향을 주지 않았지만 선형 기반 회귀 모델에는 성능 개선에 도움을 주는 것을 확인할 수 있었다.

3) 최종 후보 모델 - 캣부스트 추가

캣부스트 - 원핫인코딩	캣부스트 - 라벨인코딩
	
Test RMSE 73,806 Test MAE 43,553	Test RMSE 76,960 Test MAE 44,413
랜덤포레스트	그래디언트 부스팅
	
Test RMSE 75,483 Test MAE 44,308	Test RMSE 74,216 Test MAE 43,031

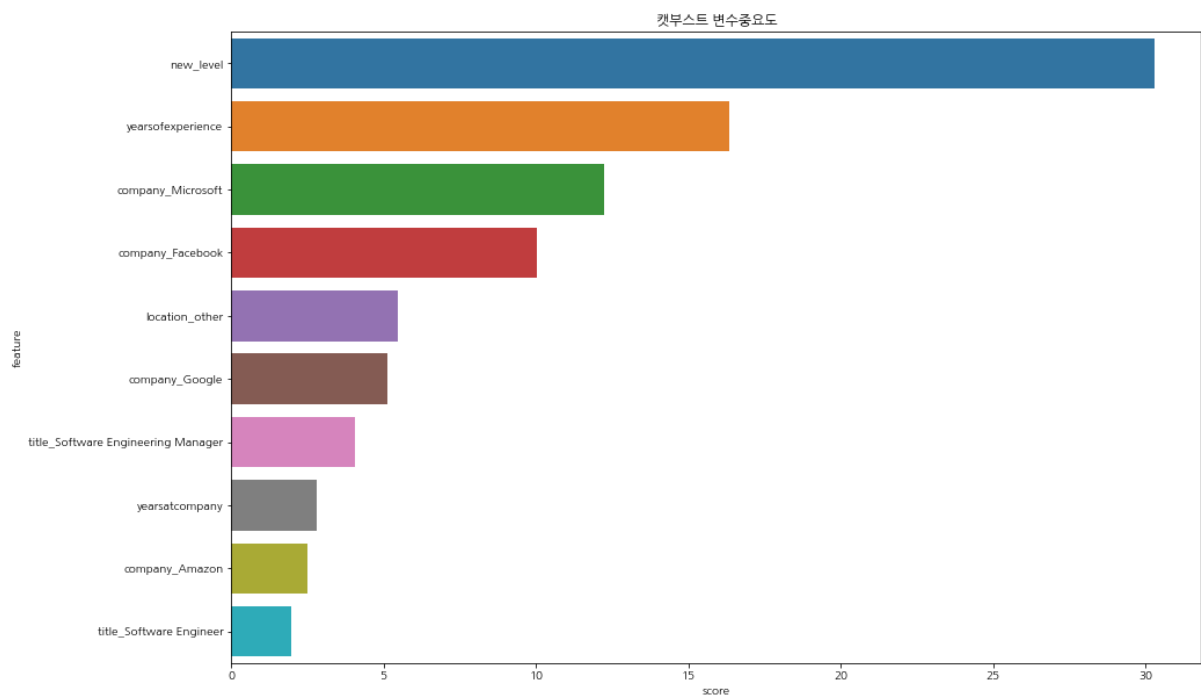
세 모델 모두 RMSE 7 만 대의 비슷한 성능을 보이며, 가장 성능이 좋은 모델은 캣부스트 모델이다. cv error 를 통해 어떤 모델이 안정적으로 좋은 성능을 보이는지 확인해보자.

모델	cv error	Mean score	Sd score
CAT	310, 321, 310, 263, 259, 253, 334, 264, 288, 274	287.99	27.55
RF	265, 330, 293, 260, 256, 256, 339, 269, 292, 276	284.02	28.42
GBR	264, 328, 262, 259, 258, 255, 340, 272, 280, 271	279.48	28.58

세 모형 모두 안정적임을 확인할 수 있으므로 test error 가 가장 좋은 캣부스트 모형을 최종 모형으로 선택한다.

IV. 최종 모형

최종 모형으로 선택한 캣부스트는 범주형 변수를 자체적인 알고리즘에 기반해 효과적으로 처리한다. 해당 데이터의 변수 정보는 경력을 제외한 나머지 8 개 변수를 범주형 변수로 볼 수 있기 때문에 트리 기반 모형에서 더 우수한 성능을 보인다고 할 수 있다. 해당 모형의 변수 중요도를 통해 중요 요인을 해석해보겠다.



변수 중요도가 높은 상위 10 개의 변수이다. 직접 전처리한 직급 변수가 가장 중요했으며, 그 다음으로 경력, 회사 정보, 직무가 중요하다. 보통 연봉은 직급에 기반하여 계약되기 때문에 해당 정보를 무시할 수 없었으며, 좀 더 확실한 전처리를 진행하거나 특정 직무 내에서만 모델링하게 된다면 더 좋은 성능을 보일 것이다.

B. Mobile phone Image data

I. 자료 설명

i. 자료 설명

이 데이터는 스마트폰 사진 데이터로, 스리랑카 전자상거래 사이트 Ikman.lk에서 스크래핑하여 캐글에 올려놓은 자료이다. 해당 스마트폰의 기종과 브랜드를 포함하고 있어 두 가지 측면에서 분류 가능하다. 중고 품목이 약 80%를 차지하며, 판매자가 직접 찍은 사진이기 때문에 스마트폰 뿐 아니라 다양한 배경이 포함되어 있다. 예를 들어 스마트폰을 들고 찍은 손이나 탁자 등 다른 물체가 사진에 포함되어 있으며 스마트폰 화면이 커져있는 경우도 있어 이러한 부분들이 학습에 방해가 될 수 있다. 또한, 어떤 사진들은 스마트폰 물건 자체가 아닌, 제품 상자를 찍은 사진이라 이를 잘 구별해내거나 학습하는 것이 중요하다.

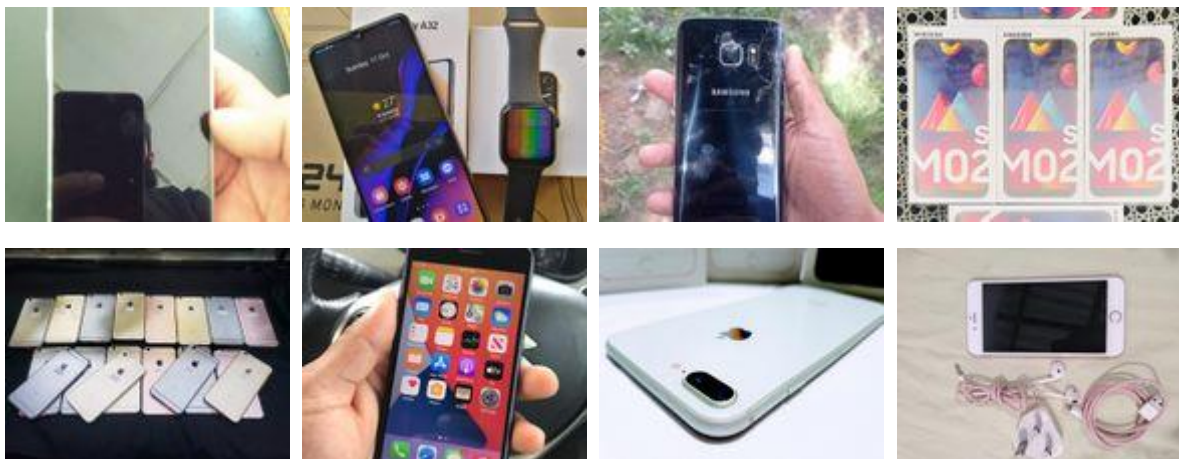
ii. 자료 출처

- 원자료 출처 : Ikman.lk

- 스크래핑된 데이터 :

<https://www.kaggle.com/lasaljwardena/mobile-smartphone-images-dataset>

iii. 이미지 예시



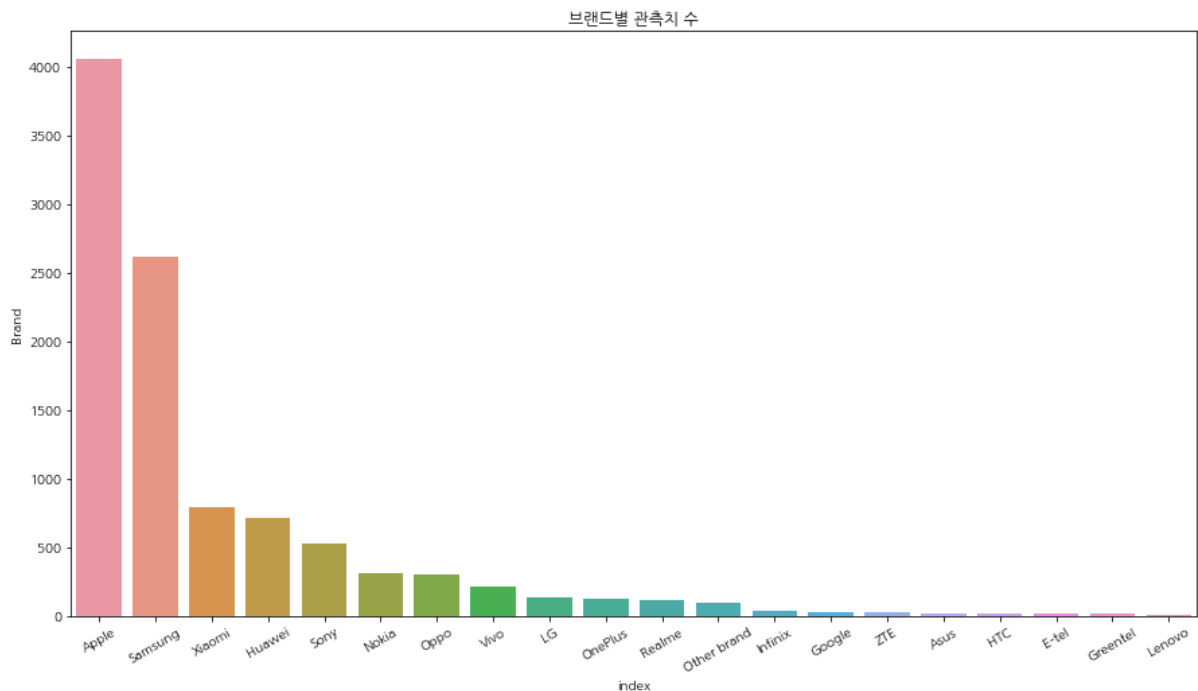
iv. 분석 목표

애플과 삼성 두 브랜드 스마트폰의 이미지를 분석하여 분류한다.

II. EDA 및 전처리

i. 반응변수

분류할 수 있는 기준으로는 크게 핸드폰 기종과 브랜드가 있다. 핸드폰 기종은 아이폰 7, 갤럭시 노트 10 등 각 핸드폰의 모델이며 총 462가지가 있다. 핸드폰 브랜드는 애플, 삼성, 샤오미 등 출시 회사이며 총 32가지가 있다. 만 개의 데이터 중 애플이 4000개, 삼성이 2500개 차지하고 있어 두 가지 브랜드로 핸드폰을 분류하는 것을 목표로 설정하였다.



ii. 전처리

원 데이터는 csv 파일에 이미지에 대한 정보와 파일 경로가 저장되어 있고, 파일 경로에 이미지가 저장되어 있다. 이를 모델링에 사용하기 위해 폴더를 생성하여 이미지를 다시 정리해주었다.

- 1) csv 파일의 이미지 경로 중 실제 이미지 파일이 없는 경우 삭제
- 2) 애플과 삼성 두 폴더를 생성하여 해당 이미지만 옮김

3) train, validation, test 데이터 분리

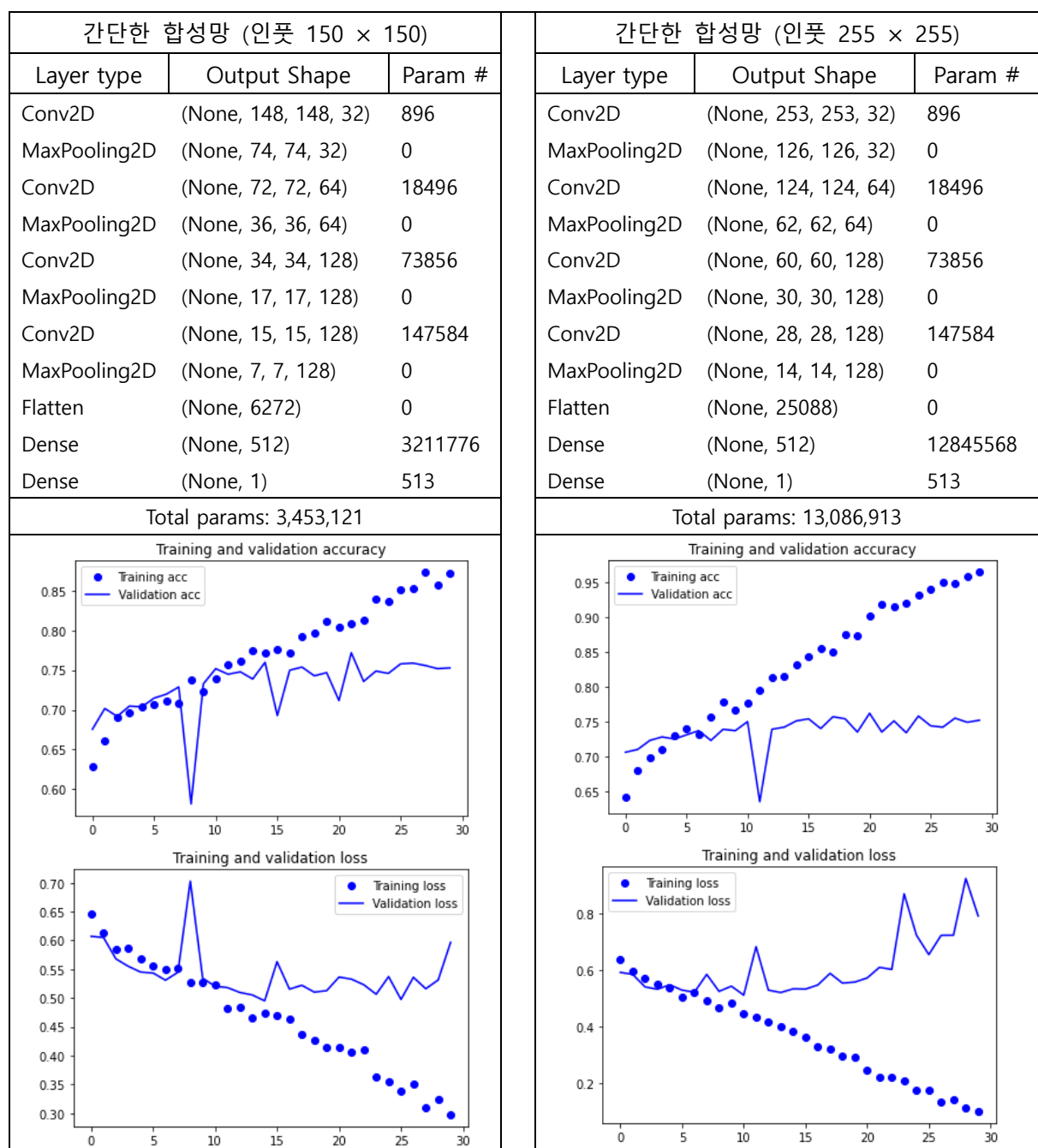
csv 파일에는 경로가 존재하나 실제로는 이미지가 없어 1번 과정을 수행해야 했다. 위 과정을 수행하는 데에는 shutil 패키지와 splitfolders 패키지를 사용하였다. Train 데이터는 70%, validation, test 데이터는 각각 15%이다. 분리 결과 디렉토리 형태는 다음과 같다.

Data_for_cnn	train	Apple
		Samsung
	val	Apple
		Samsung
	test	Apple
		Samsung

III. 모형 탐색 과정

후보 모형으로는 크게 간단한 합성망과 vgg 사전 훈련망을 사용하였고, 이미지 보강도 시도하였다. 각 후보 모형에 대해 각 이미지의 입력 shape 은 150×150 , 255×255 두 가지 버전으로 시도하였다. 이중 분류이기 때문에 loss 함수로는 binary_crossentropy 를 사용하였으며 활성화 함수로는 주로 ReLU 함수를 사용하고, 마지막 dense 층에 대해서만 시그모이드 함수를 사용하였다. 에포크는 30 으로 earlystop 없이 수렴하는지 확인하였다.

1) 간단한 합성망

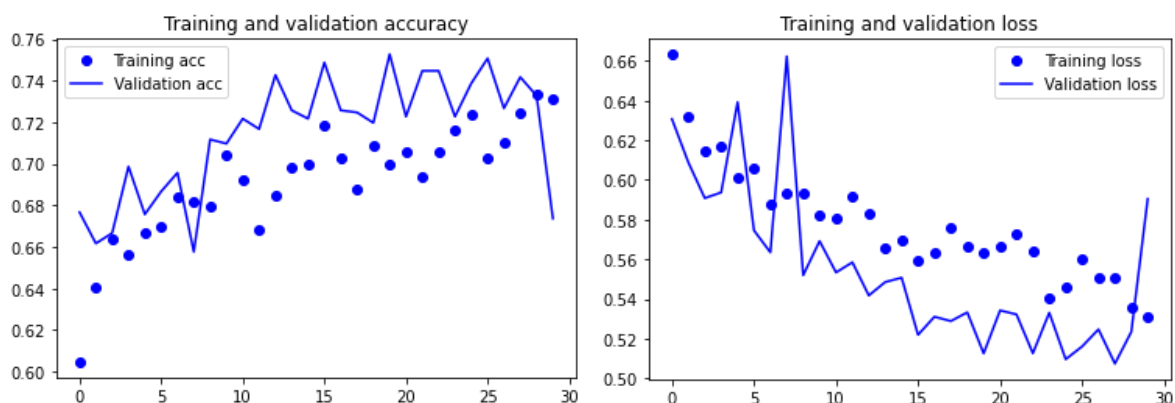


Test acc : 0.730	Test acc : 0.736
------------------	------------------

동일 모델에 대해 인풋 shape 을 달리 했을 때, 255 개로 나눈 것이 더 세밀하기 때문에 좋은 성능을 기대하였다. 하지만 학습 데이터에서 과적합이 일어날 뿐 테스트 데이터에 대한 성능 개선은 이루어 지지 않았다.

2) 이미지 보강

이미지 보강은 위의 두 결과에 차이가 별로 없었기 때문에 인풋을 150×150 로 한 모델에 대해서만 시도하였다. 이미지를 20 도 정도 회전하고, 가로 세로 약간의 차이를 준 이미지 생성기를 정의하여 과적합을 피하기 위한 시도를 하였다.



- Test acc : 0.665

결과적으로 train, val, test acc 모두 더 악화되는 성능을 보였다.

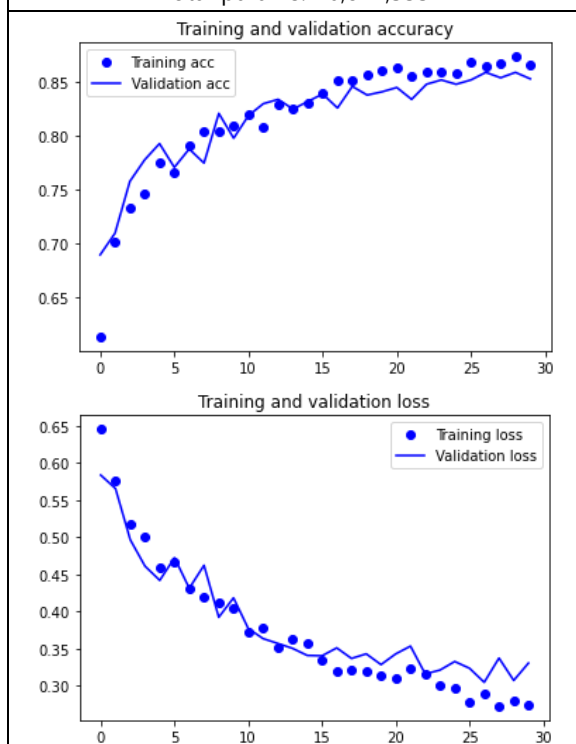
3) VGG 사전 학습망 이용

사전 학습된 VGG 모델의 다섯 번째 레이어만 본 데이터로 학습하여 사용하였다. 활성화 함수, loss 함수 등은 간단한 합성망의 설정과 동일하게 사용하였으며 입력 shape 으로는 150×150 , 255×255 두 가지를 사용하였다.

VGG 사전 학습망 훈련 (인풋 150×150)			VGG 사전 학습망 훈련 (인풋 255×255)		
Layer type	Output Shape	Param #	Layer type	Output Shape	Param #
Conv2D	(None, 150, 150, 64)	1792	Conv2D	(None, 255, 255, 64)	1792
Conv2D	(None, 150, 150, 64)	36928	Conv2D	(None, 255, 255, 64)	36928
MaxPooling2D	(None, 75, 75, 64)	0	MaxPooling2D	(None, 127, 127, 64)	0
Conv2D	(None, 75, 75, 128)	73856	Conv2D	(None, 127, 127, 128)	73856
Conv2D	(None, 75, 75, 128)	147584	Conv2D	(None, 127, 127, 128)	147584

MaxPooling2D	(None, 37, 37, 128)	0
Conv2D	(None, 37, 37, 256)	295168
Conv2D	(None, 37, 37, 256)	590080
Conv2D	(None, 37, 37, 256)	590080
MaxPooling2D	(None, 18, 18, 256)	0
Conv2D	(None, 18, 18, 512)	1180160
Conv2D	(None, 18, 18, 512)	2359808
Conv2D	(None, 18, 18, 512)	2359808
MaxPooling2D	(None, 9, 9, 512)	0
Conv2D	(None, 9, 9, 512)	2359808
Conv2D	(None, 9, 9, 512)	2359808
Conv2D	(None, 9, 9, 512)	2359808
MaxPooling2D	(None, 4, 4, 512)	0
Flatten	(None, 8192)	14714688
Dense	(None, 256)	2097408
Dense	(None, 1)	257

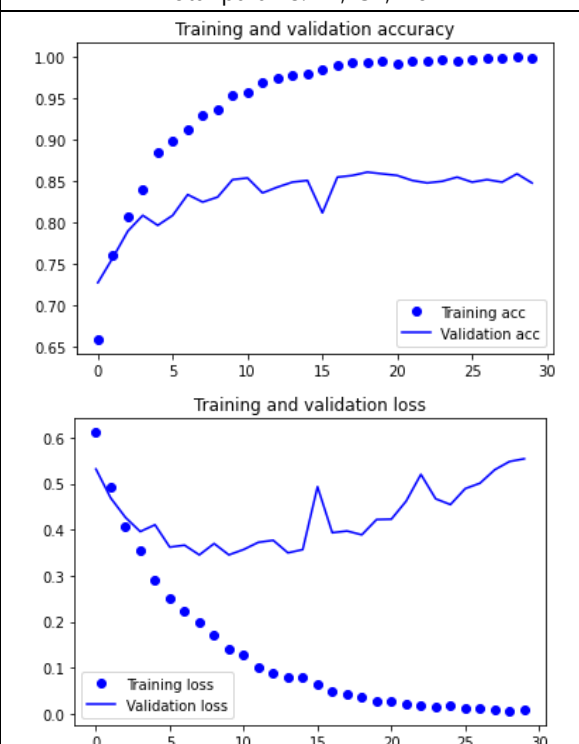
Total params: 16,812,353



Test acc : 0.851

MaxPooling2D	(None, 63, 63, 128)	0
Conv2D	(None, 63, 63, 256)	295168
Conv2D	(None, 63, 63, 256)	590080
Conv2D	(None, 63, 63, 256)	590080
MaxPooling2D	(None, 31, 31, 256)	0
Conv2D	(None, 31, 31, 512)	1180160
Conv2D	(None, 31, 31, 512)	2359808
Conv2D	(None, 31, 31, 512)	2359808
MaxPooling2D	(None, 15, 15, 512)	0
Conv2D	(None, 15, 15, 512)	2359808
Conv2D	(None, 15, 15, 512)	2359808
Conv2D	(None, 15, 15, 512)	2359808
MaxPooling2D	(None, 7, 7, 512)	0
Flatten	(None, 25088)	14714688
Dense	(None, 256)	6422784
Dense	(None, 1)	257

Total params: 21,137,729



Test acc : 0.850

사전 학습망을 이용하니 간단한 합성망보다 훨씬 성능이 개선되었다. 또한, 인풋 shape 150 × 150에서는 validation accuracy가 train accuracy를 따라가면서 과적합 또한 발생하지 않았다. 인풋 shape을 255 × 255로 늘렸을 때 오히려 과적합이 발생했으며, test accuracy가 오히려 약간 떨어지는 것을 확인하였다. 이에 drop out 층을 추가하여 보았지만 test accuracy 0.848로

개선되지 않았고, 과적합을 막을 수 없었다. 인풋 shape 150 에 대해 Data augmentation 도 사용해보았으나 마찬가지로 성능이 나아지지 않았다.

IV. 최종 모형

	간단 합성망 150	간단 합성망 255	간단 합성망 150 + 이미지 보강	VGG 사전학습망 150	VGG 사전학습망 255	VGG 255 + dropout	VGG 150 + 이미지 보강
Test acc	0.730	0.736	0.665	0.851	0.850	0.848	0.842
Epoch 당 평균 시간	33 초	166 초	70 초	265 초	847 초	757 초	281 초

VGG 사전학습망을 이용한 경우 대부분 테스트 데이터에 대한 성능이 비슷하였다. 그러나, 인풋 shape 에 따른 시간 증가가 절대적이었으며 test accuracy 와 평균 학습 시간을 고려할 때, 최종 모형은 VGG 사전학습망에 인풋 shape 을 150 으로 설정한 모형이다. 이 모형의 테스트 데이터에 대한 정확도는 85.1%이며 에포크당 265 초로 총 학습시간이 7950 초 소요되었다.

본 데이터는 스마트폰만 찍은 것이 아니라, 다양한 사물이 배경에 걸쳐 있으며, 스마트폰 화면이 커져 있는 경우가 많았다. 또한, 스마트폰은 거의 대부분 동일한 모양에 굴곡, 버튼 위치 등 약간의 디자인이 다르고, 각 브랜드마다 다양한 색상을 가지고 있기 때문에 특성을 잡아내기 어려울 수 있다. 그 결과 다양한 이미지를 학습한 VGG 사전 학습망이 본 데이터만 학습한 모형보다 성능이 훨씬 뛰어났다. VGG 이후 더 좋은 구조와 성능을 가진 모형이 많이 등장하였기 때문에 이 모형들의 사전 학습 모형을 본 데이터로 훈련시킨다면 더 좋은 결과를 얻을 수 있을 것으로 보인다.