* **분석 개요 ==========================================================================**

1. **사용 언어 :** Python
2. **분석 환경 :** [CPU] Apple M2, [RAM] 8GB, [GPU] Colab T4
3. **분석 데이터 정보**
   1. 수집 기관 : Ford 자동차 제조사
   2. 데이터 내용 : 엔진 상태와 관련 있는 센서 500개의 측정 값
   3. 데이터 파일 : Train Dataset 1file, Test Dataset 1file
4. **분석 데이터 구조**
   1. Features : 500 columns (Sensor values)
   2. Samples : 4921 raw (Train 3601, Test 1320)
   3. Labels : 1 or 0 (Normal or Abnormal)
5. **학습 및 예측 목표**
   1. 500개 각 센서 값들을 학습하여 그 상태가 Normal(1) 인지 Abnormal(0) 인지 추측
   2. Train 3601개의 샘플을 학습 후 Test 1320개 샘플의 상태를 추측
6. **데이터 학습에 사용된 모델**
   1. LogisticRegression
   2. TabularPredictor (Auto ML)
   3. XGBClassifier
   4. LGBMClassifier
   5. CatBoostClassifier
   6. RandomForestClassifier
   7. **CNN (Convolutional Neural Network)**
7. **데이터 학습 및 예측 결론**
   1. 학습이 가장 잘 되는 모델 : CNN (Convolutional Neural Network)

(정확도 97.2 ~ 97.5%, loss : 0.08 ~ 0.09)

* 1. CNN 모델은 주로 이미지나 지역적 패턴이 있는 데이터를 학습 및 예측할 때 사용되는 모델로, 분석 대상인 데이터셋도 인접 하는 센서 값과 함께 패턴을 이루는 특성이 있기 때문에 CNN 모델에서 효과적으로 학습 되는 것으로 보임
* **데이터 분석 =========================================================================**

1. 데이터를 불러와서 DataFrame 형식으로 변환

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1. train, test 데이터셋 확인

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

1. train, test 데이터셋 info()

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* + train\_df : 3601 samples, 501 columns (Include target 1)
  + test\_df : 1320 samples, 501 columns (Include target 1)
  + 500 feature type : float64, target type : object

1. Train, test 데이터셋 요약 (describe)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

1. Target value 1과 0으로 변환

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

1. Target value 빈도의 균형이 맞는지 확인

텍스트, 스크린샷, 소프트웨어, 도표이(가) 표시된 사진

자동 생성된 설명

* + train, test dataset 의 target value 빈도는 균형이 맞다.

1. 0과 1 클래스 각 샘플의 센서 값 분포 비교 (각 3개 샘플 비교)

텍스트, 스크린샷, 폰트, 라인이(가) 표시된 사진

자동 생성된 설명

* + 클래스별 센서 값에 차이가 있음 -> 학습 가능할 것으로 보임
  + 전체 샘플의 센서 별 평균도 볼 필요 있음 (클래스 구분)

1. 센서 별 클래스 평균 값 분포

텍스트, 스크린샷, 폰트, 소프트웨어이(가) 표시된 사진

자동 생성된 설명

도표, 그래프, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* + 센서 후반부 (s403 ~) 클래스에 따라 센서 값이 양수와 음수로 나눠짐 -> 후반부 센서만 학습시켜 볼 필요 있음

1. 전체 샘플에 대한 센서 값들의 분포 (클래스별)

텍스트, 스크린샷, 라인, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 각 샘플 값이 시계열 데이터일 경우 시간의 흐름에 따른 패턴은 보이지 않음
  + 클래스별 평균 값으로 다시 시각화 해볼 필요 있음 (다음 장)

1. 각 샘플 별 500개 센서 값의 평균

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

* + 출력 결과 다음 페이지 참조

스크린샷, 텍스트, 라인, 도표이(가) 표시된 사진

자동 생성된 설명

* + 0클래스에 노이즈 샘플이 있음.
  + 노이즈 샘플 제거 후 학습 해볼 필요 있음.

1. 센서 별 상관관계 분석

텍스트, 스크린샷, 다채로움, 디자인이(가) 표시된 사진

자동 생성된 설명

인접한 센서와 상관계수가 높음

* CNN으로 학습 가능 할

것으로 보임

* kernel\_size를 3으로하면

학습이 잘될 것으로 예상

1. Target에 영향을 주는 센서 확인

텍스트, 스크린샷, 디자인이(가) 표시된 사진

자동 생성된 설명

* + 앞서 확인 한대로 후반부(s403 ~)에 target에 영향을 주는 센서가 있음.
  + Target과 상관계수가 높은 센서들만 학습해볼 필요 있음.

=================================================================================

* + train, test dataset 라벨 분리 (x, y)

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

* **학습 및 결과 예측 ===================================================================**

1. Auto ML(AutoGluon)을 사용하여 최적의 머신러닝 모델 확인 및 평가하기

텍스트, 스크린샷, 문서, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 최적의 모델로 학습 및 평가 결과

텍스트, 스크린샷, 도표, 라인이(가) 표시된 사진

자동 생성된 설명

* + Best model : WeightedEnsemble\_L2
    - Accuracy: 84.7%
    - Precision: 84.41%
    - Recall: 83.88%
    - F1 Score: 84.14%
    - AUC: 91.95%
    - threshold 0.53 -> Accuracy : 85.08%
  + Leaderbord 결과 보기

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 학습 모델 14가지 중 WeightedEnsemble\_L2 모델이 가장 성능이 좋음

1. 머신러닝 4가지 모델 학습 및 결과 예측

텍스트, 스크린샷, 문서, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트, 문서이(가) 표시된 사진

자동 생성된 설명

* + 모델별 평가 결과 (베스트 모델 : CatBoost 84.55%)

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명 텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

* + 모델별 혼동행렬, ROC Curve

텍스트, 스크린샷, 도표, 평행이(가) 표시된 사진

자동 생성된 설명

* + 혼동 행렬의 4가지 항목 (TP, TN, FP, FN) 모두 CatBoost 성능이 가장 좋음
  + Feature\_importances\_ : 각 특성의 중요도 (모델의 예측에 얼마나 기여했는지)
  + Featrue\_importances\_ 값은 target 상관관계 값과 연관성 적음
    - 모델별 상위 10개 센서와 target 상관관계 상위 센서와 연관성 적음

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명텍스트, 스크린샷, 폰트, 그래프이(가) 표시된 사진

자동 생성된 설명

1. Logistic Regression 모델 학습 및 결과 예측

텍스트, 스크린샷, 문서, 폰트이(가) 표시된 사진

자동 생성된 설명

* + Logistic Regression 평가 결과

텍스트, 도표, 스크린샷, 그래프이(가) 표시된 사진

자동 생성된 설명

* + Logistic Regression metrics
    - Accuracy: 49.55%
    - Precision: 47.8%
    - Recall: 45.85%
    - F1 Score: 46.81%
    - AUC: 48.7%
    - threshold 0.35 -> Accuracy : 51.14%
  + 정확도, AUC 50% 미만으로 Logistic regression 모델로는 학습이 안되는 것으로 나타남

1. CNN 학습 및 결과 예측

텍스트, 스크린샷, 문서이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 문서, 메뉴이(가) 표시된 사진

자동 생성된 설명

* + CNN 모델 평가 결과

[테스트데이터의 정확도, loss]

test acc 0.9734848737716675

test loss 0.08908886462450027

[CNN Model metrics]

- Accuracy: 97.35%

- Precision: 97.04%

- Recall: 97.5%

- F1 Score: 97.27%

- AUC: 99.43%

[훈련데이터, 검증데이터의 정확도, loss 그래프]

텍스트, 스크린샷, 그래프, 라인이(가) 표시된 사진

자동 생성된 설명

* + CNN 모델의 혼동행렬과 ROC Curve

(출력 코드는 머신러닝 코드와 동일하여 생략)

도표, 텍스트, 라인, 그래프이(가) 표시된 사진

자동 생성된 설명

* + CNN 모델 정확도 97.2 ~ 97.5%, loss 0.08~0.09, AUC = 0.99 로 학습이 잘 됨
  + CNN 모델 하이퍼파라미터 튜닝

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명

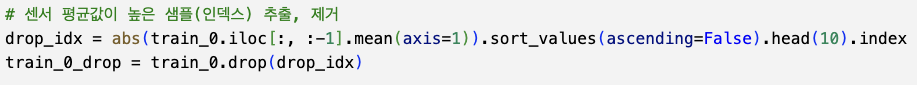
* **Conv 1D Layer** 
  + 최적 조건 : 3 Layer
  + 2Layer 또는 4Layer일때 정확도 떨어짐(95% 이하), 5Layer일때 과적합 발생
* **활성 함수**
  + 최적 함수 : LeakyReLU
  + Relu, swish, softplus에 비해 과적합이 덜 일어나고, 정확도도 가장 좋게 나타남
* **Optimizer**
  + 최적 optimizer : adam
  + SGD 일때 400epoch기준 정확도 떨어짐(92%), 2000epoch 까지 늘려도 97%에 도달하지 못함.
* **필터 수**
  + 최적 필터 수 : 30 ~ 32
  + 필터 수가 많으면 특징은 더 많이 추출하겠지만, 많을수록 성능이 좋아지는 것은 아님. 최적의 필터 수는 실험 결과 30 ~ 32가 가장 적합함.
* **Batch\_size**
  + 최적 배치 수 : 16~24
  + 배치 수가 많을 수록 학습 속도는 빨라지나, 정확도가 떨어지는 경향이 있음
  + 실험 결과 과소적합, 과적합이 일어나지 않으면서 정확도나 loss 값이 가장 잘 나오는 조건은 16~24
* **최적모델 저장 기준(monitor)**
  + 최적 기준 : ‘val\_sparse\_categorical\_accuracy’
  + ’val\_loss’ 를 기준으로 할 경우 정확도가 대체로 0.5% 정도 떨어짐
* **검증 데이터 비중**
  + 최적 비중 : 15~20% 일 때 성능이 가장 좋게 나타남
* **Kernel\_size**
  + 최적 커널 수 : 3
  + 커널 수가 2, 4, 5일 경우 정확도 떨어짐 (94% 이하)
  + 인접 센서와 연관성이 높아 길이 3의 패턴이 의미 있는 특성을 가지고 있는 것으로 보임
* **데이터 정규화(StandardScaler)**
  + 최적 조건 : 데이터 정규화 불필요
  + 데이터 정규화를 진행했을 경우 정확도가 비슷하거나 약간 낮아짐 (효과 없음)
* **배치 정규화(BatchNormalization)**
  + 최적 조건 : 배치 정규화 필요
  + 배치 정규화 안 했을 경우 성능 크게 떨어짐 (92% 이하)
* **Epoch**
  + 최적 Epoch : 400회
  + Epoch 400회 초과 되어도 성능이 더 이상 개선 되지 않음

(단, SDG optimizer 사용했을 경우는 예외)

1. 데이터 전처리 후 재학습(1)

--- 500개 전체 센서의 평균 값이 높은 샘플(노이즈) 데이터 제거 후 학습

* 1. 평균 값 상위 10개 샘플 제거 코드



* 1. 상위 10개 샘플(노이즈) 제거 후 시각화

[노이즈 제거 전]

[노이즈 제거 후]

텍스트, 스크린샷, 라인, 도표이(가) 표시된 사진

자동 생성된 설명 스크린샷, 텍스트, 도표, 디자인이(가) 표시된 사진

자동 생성된 설명

* 1. 상위 5, 10, 20개 샘플 제거 후 학습 및 예측 결과

🡪 성능 개선 되지 않음

* + - 머신 러닝 모델 : 예측 정확도 비슷하거나 약간 낮은 수준
    - CNN 모델 : 예측 정확도 비슷한 수준

1. 데이터 전처리 후 재학습 (2)

--- S403 센서 이후 데이터만 학습

텍스트, 스크린샷, 폰트, 디자인이(가) 표시된 사진

자동 생성된 설명

* + 403번 센서부터 0클래스와 1클래스의 평균 값이 음수와 양수로 명확히 갈리는 현상. 이부분이 target에 영향을 크게 준다고 추측 할 수 있음
  + 403번 이후 센서 데이터만 학습하여 예측 해보기 (다음 페이지)
* s403 ~ s500 데이터셋, 라벨 추출하기

텍스트, 폰트, 소프트웨어, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* s403 ~ s500 센서만 학습하여 예측한 결과

🡪 성능 개선 되지 않음 : 데이터 부족으로 과소적합 발생되는 듯

* + 머신 러닝 모델 : 예측 정확도가 모델별로 2~10% 감소
  + CNN 모델 : 예측 정확도 89~90%로 7% 정도 감소 (loss는 0.09에서 0.25로 증가)

1. 데이터 전처리 후 재학습 (3)
   1. Target과 상관계수 값이 높은 센서로만 학습
   2. 상관 계수 낮은 센서 drop (10개, 30개 drop이후 학습 및 예측)’
   3. target과 상관계수 낮은 센서 drop 시키는 코드

텍스트, 스크린샷, 폰트, 웹 페이지이(가) 표시된 사진

자동 생성된 설명

* 1. taret과 상관계수 낮은 센서는 제거 후 학습 및 예측 결과

🡪 성능 개선 되지 않음

* + 머신 러닝 모델 : 모델별로 예측 정확도 비슷하거나, 1~2% 정도 떨어짐
  + 딥러닝 모델 : 하위 10개 제거 시 정확도 2% 정도 떨어짐, 하위 30개 제거 시 과소적합 발생 (정확도 90%, loss 0.24)

🡪 인접 하는 센서와 함께 특징을 구성하지만 중간 중간 데이터가 빠질 경우 특징을 추출하는데 악영향을 주는 것으로 추측

* **데이터 분석 및 학습 실험 결과 =====================================================**

1. 최적의 학습 모델
   * CNN (Convolutional Neural Network)
   * 정확도 : 97.2 ~ 97.5 %
2. 데이터 전처리 후 학습 결과 (3가지)
   * 효과 없음

(1) 평균 센서 값 높은 샘플 제거 (5개, 10개, 20개 행 제거)

(2) 403번 이후 센서만 학습

(3) target 상관관계 하위 센서 제거 (10개, 30개 열 제거)

1. 모델별 학습 결과 요약

텍스트, 스크린샷, 번호, 폰트이(가) 표시된 사진

자동 생성된 설명