2024.08.26 머신러닝 실습 보고서

Created	@2024년 8월 26일 오후 3:47
Class	8.19~8.26 머신러닝

이름	곽태경
소속	빅데이터 9기

요약

8월 26일 실습에서는

학습 사이트 서비스를 이용하는 사람들에 대한 데이터에서 사용자의 행동 변화, 회원 상태 변화 데이터들을 사용해서, 사용자의 행동 패턴을 학습해서 소프트 보팅 방식으로 사용자의 다음 행동 패턴 예측값들을 산출해 보고, 검증해보는 과정도 거쳤습니다.

도입

문제 정의

서비스를 사용하는 대부분의 사용자들은 본인들만의 행동 패턴이 있기 마련이고, 익숙한 선택지를 골라 행동할 것입니다. 만약 사용자에게 서비스를 제공할 때, 사용자의 다음 행동을 예측할 수 있다면, 사용자에게 익숙한 선택지를 줄 수 있고, 사용자에게 맞는 더 적합한 서비스를 제공할 수 있을 것입니다.

기획 의도

- 사용자의 행동 패턴을 분석하고 머신 러닝 모델에 학습시켜 사용자의 다음 행동을 예측하기 위해서입니다.

개발 환경

- Vscode 1.92.2
- Anaconda 1.12.3
  - Python 3.12.4
  - Numpy 1.26.4
  - Pandas 2.2.2
  - matplotlib 3.8.4
  - Scikit_learn 1.4.2
  - Seaborn 0.13.2
  - jupyter notebook v2024.7.0

데이터 정보

파일명 : activity_sample.xlsx

데이터 정보

분류	값(개수)
sample	6476
features	44

features 목록

features	정보	데이터 타입	데이터 예
userid	사용자 id	object	01cf7916-eb98-4d30-8c81-dd52aa5be060
learning_seq	학습 순서	int64	1
mcode	콘텐츠 고유 id	object	T0TE52U01003
learning_action_seq	학습 중 행동 순서	int64	2
event_type	이벤트 유형	object	ViewEvent
action	이벤트에 포함되는 행동 내용	object	Viewed
object_type	활동 대상	object	DigitalResource

timestamp	해당 학습 중 행동이 발생한 시간	object	2022-09-14 23:28:53.150
gender	사용자의 성별	object	M
grade	사용자의 학년	int64	5
memberstatus	회원 상태(정회원, 준회원, 탈퇴회원 등)	int64	44
memberstatus_change	월 중 회원 상태 변화	object	11,44
'day_01_status' ~ 'day_31_status'	1일 ~ 31일 회원 상태 (총 31개 열)	1~12 : object, 13~31 : int64	11 / 44
change_date	회원 상태 변화까지 걸린 시간	int64	25

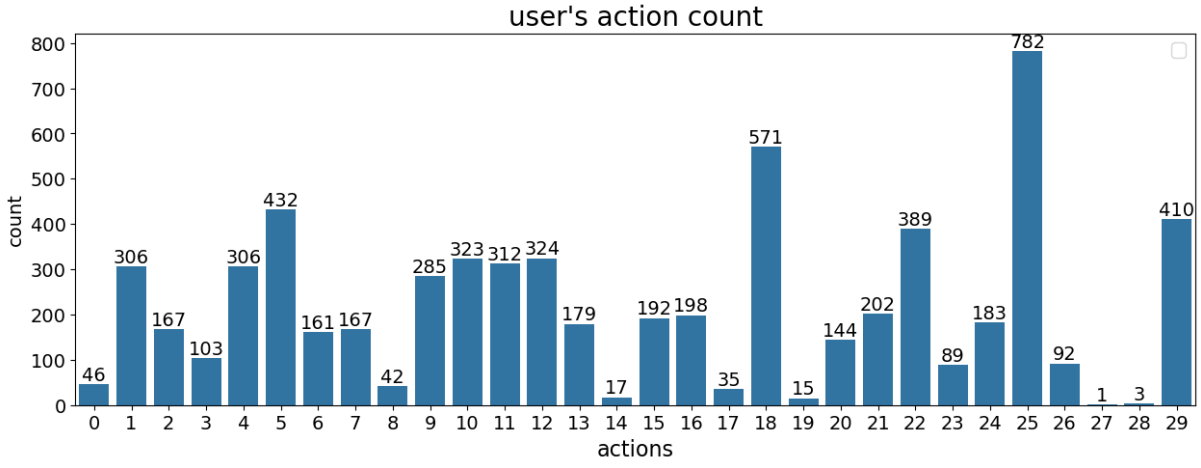
개발 과정

1. EDA

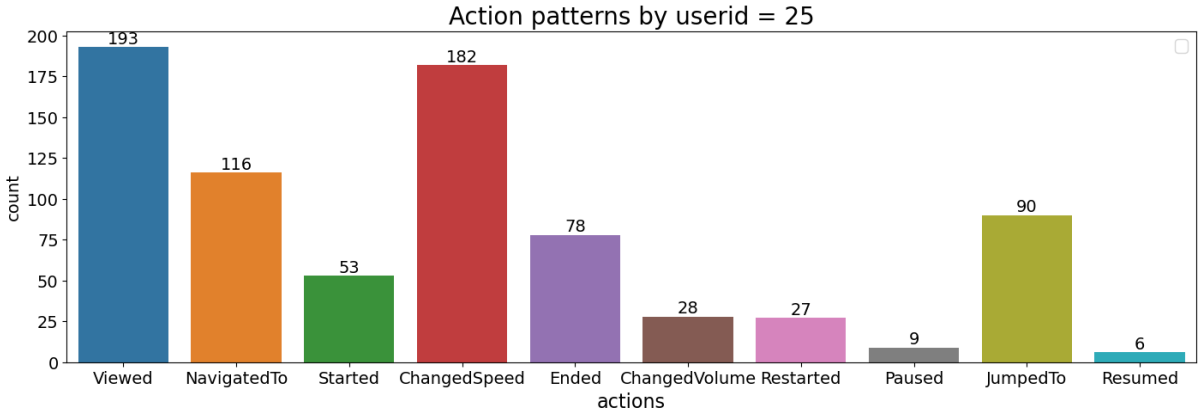
'userid', 'mcode'

이 feature들은 데이터 타입이 코드 형태의 object인 범주형 feature들인데, 정수형으로 만들어 모델에 학습시키기 쉽게 하기 위해 LabelEncoder로 인코딩하여 정수형으로 된 범주형 데이터로 만들었습니다.

그리고 사용자들의 행동 패턴 개수를 시각화 해 보았습니다.



여기서 user = 25인 사용자가 행동 패턴 개수가 782개로 가장 많았기에, user = 25인 사용자의 행동 패턴들의 가짓수도 시각화해 보았습니다.



그래프들을 보고, 가장 행동 패턴 개수가 많은 user = 25인 사용자(이하 25번 사용자)의 다음 행동 패턴을 예측하는 머신 러닝 모델을 구현하기로 했습니다.

이용하기로 한 feature는 (모두 25번 사용자의 데이터만을 사용했습니다.)

learning_seq	학습 순서
learning_action_seq	학습 중 행동 순서
mcode	콘텐츠 고유 id
object_type	활동 대상

이 4가지를 이용하기로 했습니다. 여기서 object_type도 feature로 이용하기 위해서, LabelEncoder로 인코딩하여 정수형으로 된 범주형 데이터로 만들었습니다.

그리고 target은

action	이벤트에 포함되는 행동 내용

을 사용하여 feature들로 target을 예측하는 머신 러닝 모델을 설계했습니다.

action은

- Viewed
- ChangedSpeed
- NavigatedTo
- JumpedTo
- Ended
- Started
- ChangedVolume
- Restarted

- Paused
- Resumed

총 10개의 범주로 이루어진 feature입니다.

2. 모델 학습

학습용 샘플과 테스트용 샘플을 나눴는데, 일단 'timestamp'와 'learning_action_seq'를 기준으로 오름차순 정렬하여 일어난 순서대로 정렬한 후, 순서를 유지시키기 위해 직접 데이터프레임을 8:2로 슬라이싱해서 나누어 학습용 샘플과 테스트용 샘플을 만들었습니다.

(학습용 샘플 : 625개, 테스트용 샘플 : 157개)

그리고 머신러닝 모델은 RandomForestClassifier, LogisticRegression, GradientBoostingClassifier,

SVC, KNeighborsClassifier를 모두 사용해 소프트 보팅 방식(모든 모델의 클래스 확률을 평균내어 가장 높은 평균 확률을 가진 클래스를 최종 예측으로 선택하는 방식)을 통해 최종 예측 결과를 산출하는 모델을 만들었습니다.

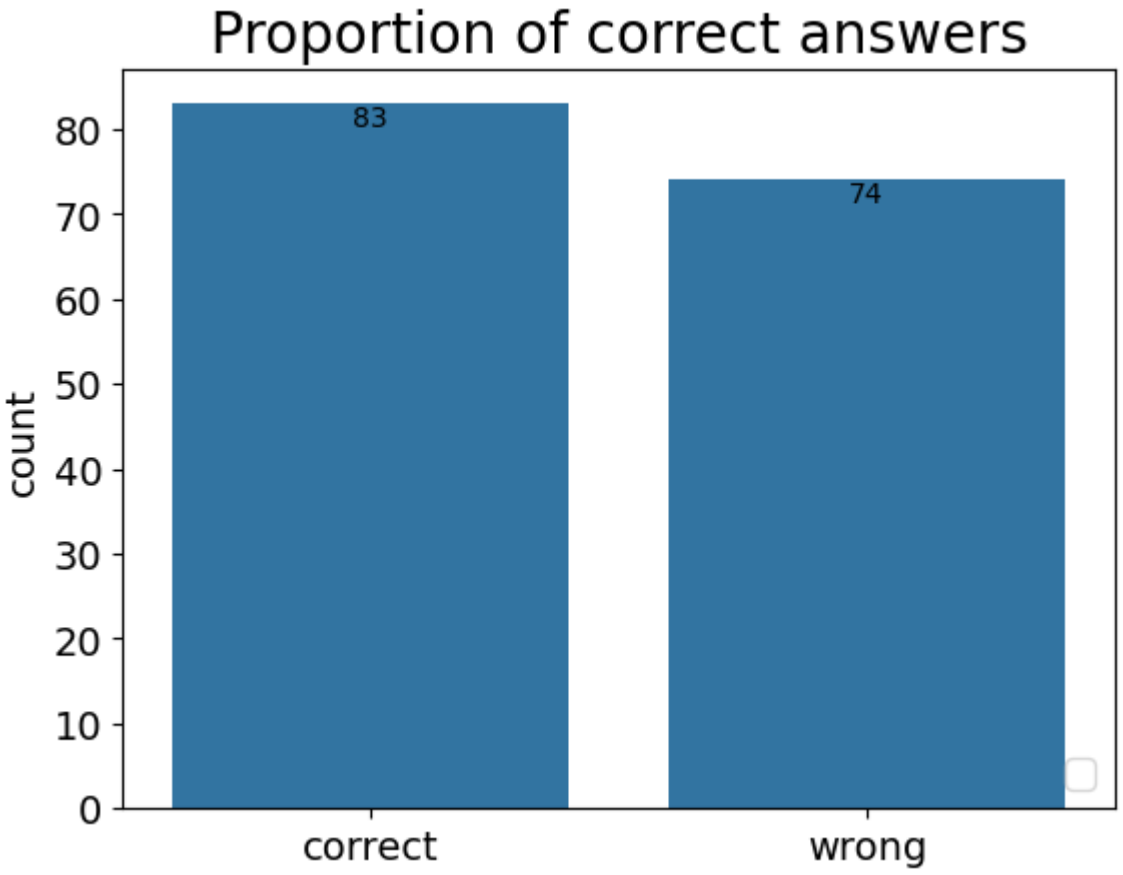
하이퍼파라미터	값	들어간 모델명
n-estimators(결정 트리 개수)	1000	RandomForestClassifier, GradientBoosingClassifier
probability(확률 계산)	True	SVC

그리고 25번 사용자의 다음으로 할 행동의 예측값을 산출하고, 테스트 target의 데이터와 비교해보았습니다.

순서	action(real)	action(predict)	예측 성공 여부
1	NavigatedTo	NavigatedTo	correct
2	Viewed	Viewed	correct
3	NavigatedTo	NavigatedTo	correct
4	Viewed	Viewed	correct
5	ChangedSpeed	ChangedSpeed	
…	…	…	…
153	JumpTo	Ended	wrong
154	Ended	Resumed	wrong
155	NavigatedTo	Viewed	wrong
156	Viewed	NavigatedTo	wrong

157	Viewed	NavigatedTo	wrong

그리고 테스트 샘플 전체의 정답과 오답 비율을 시각화 해 보았습니다.



그래프를 보면, 전체 157개 샘플 중 틀린 갯수가 74개나 되는 것으로 보아 모델의 학습이 제대로 이루어지지 않는다고 판단하고, 모델 검증 과정을 실행했습니다.

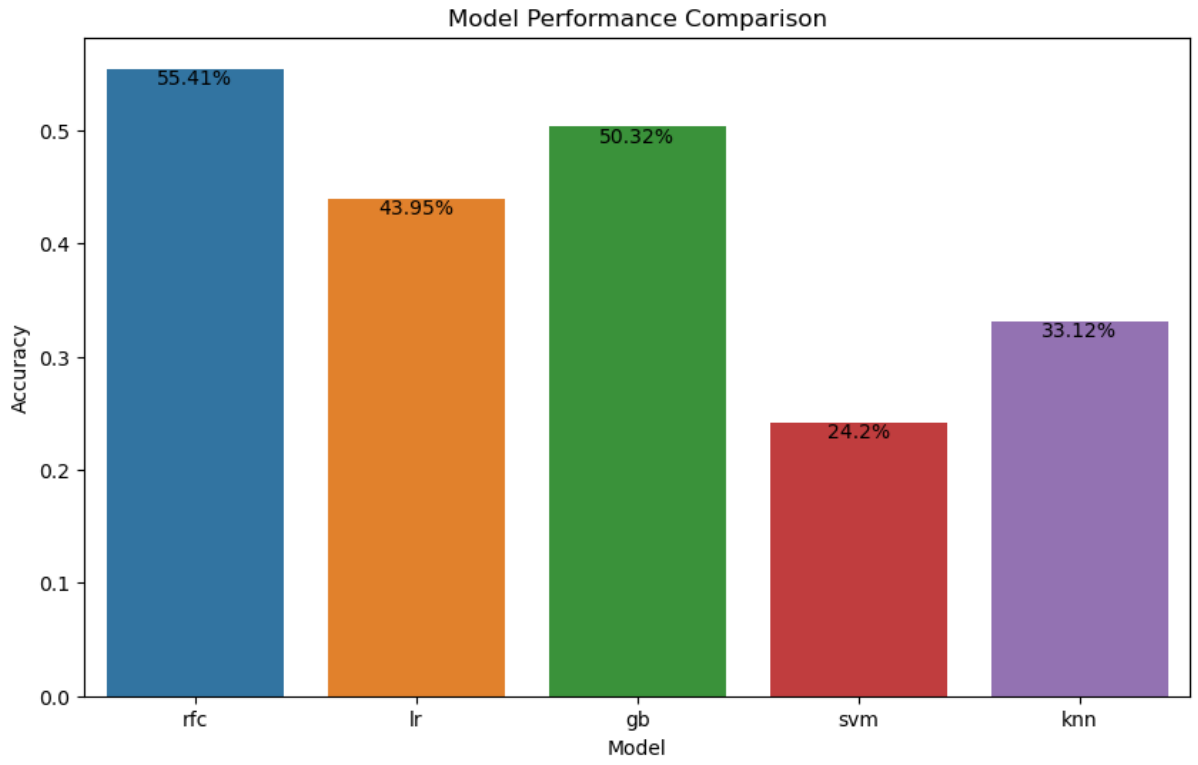
3. 모델 검증

학습 데이터 정확도와 테스트 데이터 정확도를 계산했습니다.

학습 데이터 정확도 : 0.9856

테스트 데이터 정확도 : 0.5287

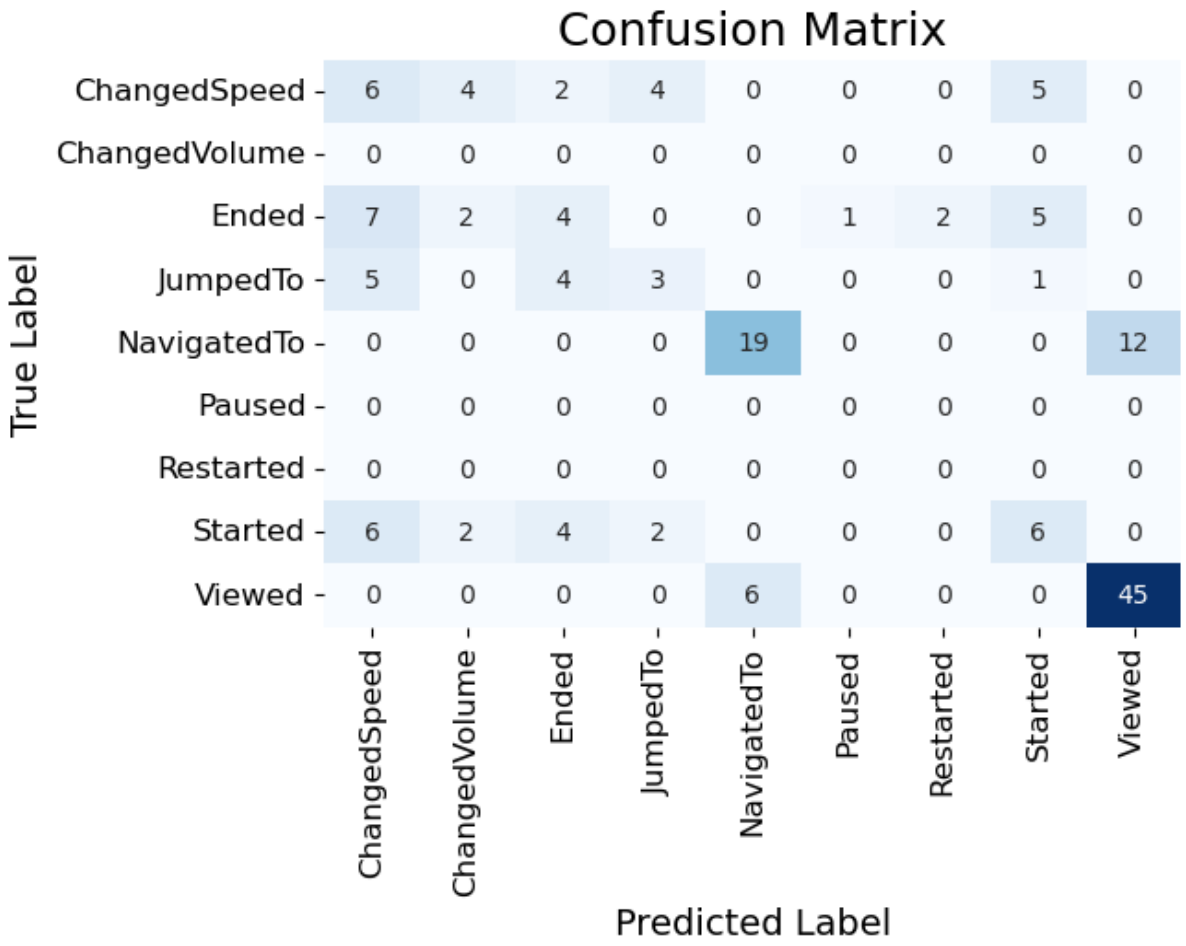
사용한 모델 별 테스트 샘플 정확도도 시각화해보았습니다.



사용한 5개 모델 중에서는 RandomForestClassifier가 가장 정확도가 높았습니다.

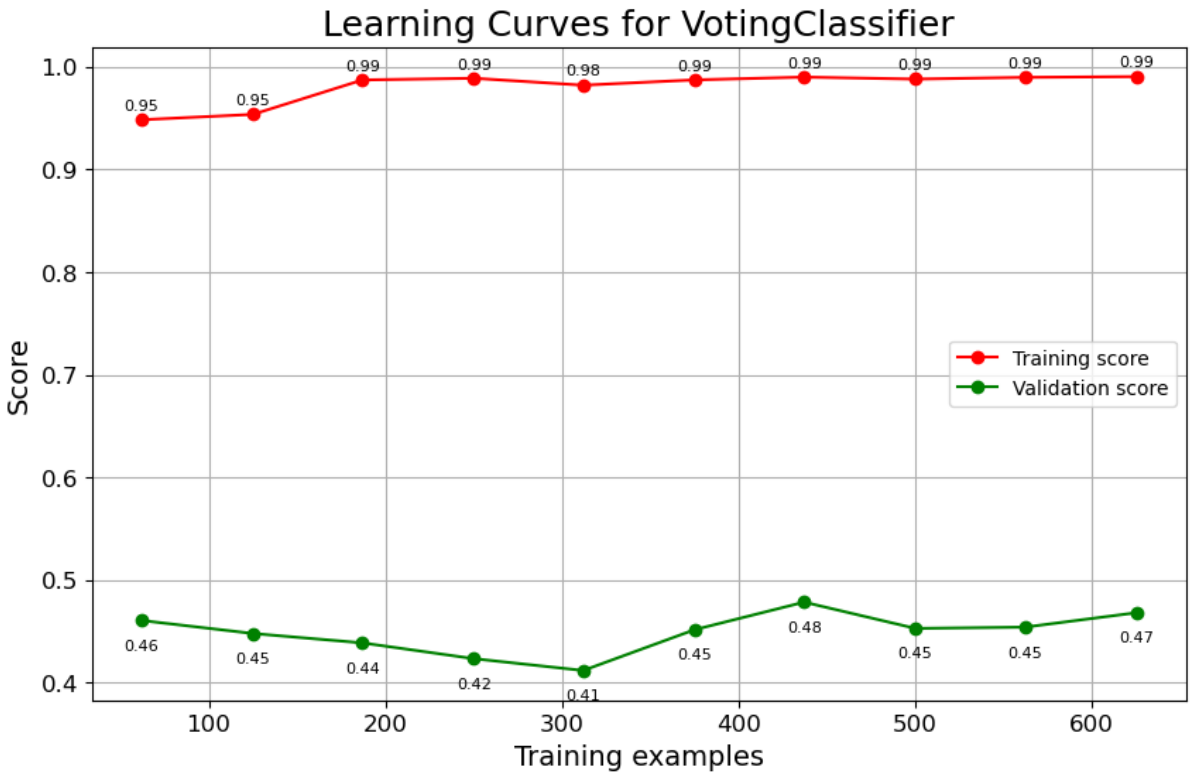
그리고 혼동 행렬을 confusion_matrix 메서드로 계산했고, heatmap으로 시각화했습니다.

('Resumed' 클래스는 테스트 샘플에서도 한 번도 나오지 않았기 때문에 히트맵에 존재하지 않습니다.)



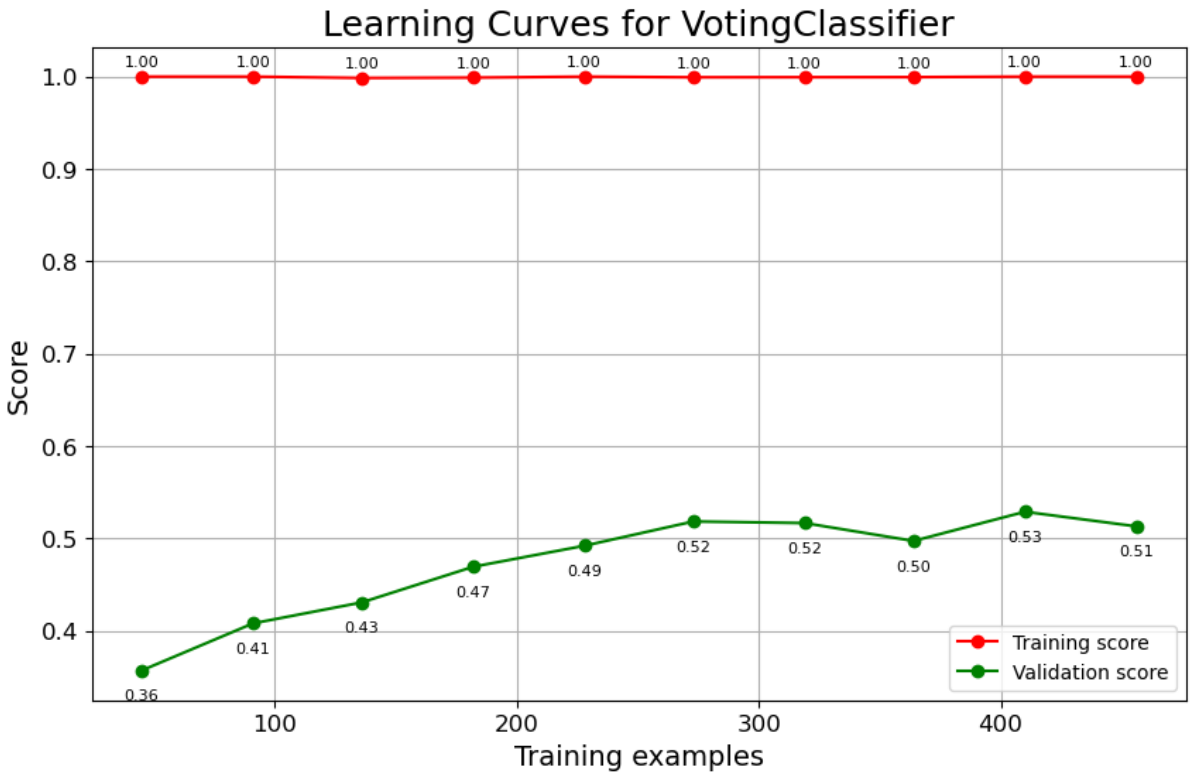
학습 데이터와 테스트 데이터 정확도를 보니, 학습 데이터 정확도는 98%지만 테스트 데이터 정확도는 52%로 훨씬 못 미치는 것으로 판단했습니다.

혼동 행렬도 'Viewed' feature를 제외하면 한 feature에 최대 6가지로 나누어진 예측 답안을 내놓은 것을 보고, 이 모델은 과적합이 의심되어 학습 곡선을 그려 보았습니다.



학습 곡선을 보니, 두 그래프의 모양이 평행하고, 학습용 샘플의 정확도는 매우 높지만, 테스트용 샘플의 정확도가 학습용 샘플의 정확도에 훨씬 못 미치는 모습을 보여 이 모델은 과적합인 것으로 판단하였습니다.

더 적은 샘플(571개)을 가진 18번 사용자의 샘플로 학습 곡선을 다시 그려 보았는데,



학습용 데이터 샘플의 정확도가 100%에 달해 과적합이 더 심해진 것으로 보이고, 과적합을 해결하려면 더 많은 양의 샘플을 사용해야 한다고 판단하였습니다.

결론

이번 실습에서는 25번 사용자의 다음 행동 패턴을 예측해 보았습니다. 5가지의 머신러닝 모델을 사용하여 소프트 보팅 방식으로 예측값을 산출해 보았습니다. 그런데 학습용 샘플 정확도는 98%가 넘었지만, 테스트용 샘플 정확도는 52%로 그에 훨씬 못 미치는 정확도를 보였고, 혼동 행렬에서도 'Viewed'를 제외한 클래스에서 6개 이상으로 예측값이 나뉘는 것을 볼 수 있었고, 학습 곡선이 평행할 뿐만 아니라 높은 학습용 샘플 정확도에 비해 테스트 샘플의 정확도가 낮은 그래프 양상을 보여 이는 과적합일 확률이 굉장히 높다고 판단하였습니다. 샘플이 더 적은 사용자일수록 과적합이 심해지는 것으로 보아, 샘플 개수가 부족한 데 비해 클래스의 개수가 너무 많아서 일어나는 현상으로 보여집니다. 결론적으로, 이 모델의 학습도를 높이려면, 더욱 복잡한 머신러닝을 수행할 수 있는 딥러닝 모델을 사용하는 것이 좋은 방안이 될 수 있을 것입니다.

개선점

- 소프트 보팅 방식에서 모델을 5가지를 사용했지만, 그렇다고 해서 모델 각각의 정확도에 비해 보팅 모델의 신뢰도가 높아지는 모습은 보이지 않았습니다. 모델들의 특성들을

고려해서 정확도가 높은 모델들만 모아서 묶거나, 정확도가 가장 높은 모델 하나만 사용해서 학습시키는 것이 더 좋은 결과를 가져올 것으로 보입니다.

- 무려 10종류가 되는 행동 패턴 종류를 생각했을 때, 샘플의 개수 부족 문제 뿐만 아니라 이 예측 모델을 구현하려면 단순한 머신러닝 모델로는 역부족이라고 생각됩니다. 그래서 만약 설계한 대로 동작하게 하려면, 딥러닝 모델을 사용해야 할 것으로 보입니다.

소스 코드

보고서 제출 시에 같이 첨부하겠습니다.

파일 이름 : 8.26_실습.ipynb

데이터셋 'activity_sample.xlsx'를 소스 코드 파일과 같은 폴더에 넣고 소스 코드를 실행시키시면 됩니다. (데이터셋 파일 경로는 ./activity_sample.xlsx)

참고 문헌

수업에서 배운 내용을 토대로 실습했습니다.