제주도 관광객 카드 소비금액 예측하기

19011494 조국희

제주도 날씨 및 관광객 정보에 따른 카드 소비금액 예측하기

선정이유

- ✓ 관광객들의 특성 및 지역의 날씨에 따라 소비 금액이나 형태는 달라지기 때문에 해당 데이터들의 학습을 통해 관광객들의 소비 예측이 가능할 것
- ✓ 이러한 예측을 통해 관광객들을 상대하는 자영업자들은 이들의 소비 성향을 파악하고 미리 대비할 수 있음
- ✓ 데이터의 학습 결과 분석을 통해 변화하는 관광객들의 소비트렌드 파악 가능

데이터 수집

- ✓ 제주도 관광객의 개인 정보에 따른 카드 소비금액 데이터

  https://www.data.go.kr/data/15046091/fileData.do

- ✓ 제주도 지역별 평균 강수량

  https://data.kma.go.kr/stcs/grnd/grndRnList.do#

- ✓ 제주도 지역별 기온

  https://data.kma.go.kr/stcs/grnd/grndTaList.do?pgmNo=70

기상청 기상자료개방포털

데이터 수집

2014년 9월부터 2016년 8월까지의 월 별 기온 및 강우량, 관광객들의 정보를 포함한 총 11개의 feature

- ✓ **기준년월** : 관광객의 소비금액이 측정된 년월

관광객 정보

- ✓ 제주 대분류 : 제주도 내 시
- ✓ 제주 중분류 : 제주도 내 동, 읍
- ✓ 업종명 : 관광객이 카드를 사용한 업종
- ✓ 성별 : 관광객의 성별
- ✓ 연령대 : 관광객의 연령대
- ✓ 카드이용건수 : 관광객의 월 카드이용건수

날씨 정보

- ✓ 평균 강수량 : 월 평균 강수량
- ✓ 평균 기온 : 카드 이용장소의 월 평균 기온
- ✓ 최저 기온 : 카드 이용장소의 월 최저 기온
- ✓ 최고 기온 : 카드 이용장소의 월 최고 기온

데이터 병합

관광객 정보 data

	A	B	C	D	E	F	G	H	I	J	K	L
1	기준년월	관광객 유형	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용금	카드이용건	건당이용금	데이터기준일자	
2	Sep-14	내국인 관광	서귀포시	대륜동	농축수산품	여	50대	14434000	67	215433	######	
3	Sep-14	내국인 관광	서귀포시	대륜동	농축수산품	남	50대	15119000	72	209986	######	
4	Sep-14	내국인 관광	서귀포시	대륜동	농축수산품	여	40대	7609500	41	185598	######	
5	Sep-14	내국인 관광	서귀포시	남원읍	농축수산품	남	50대	7092500	40	177313	######	

```
import numpy as np
import pandas as pd
X = pd.read_csv('card.csv', encoding='cp949').drop(['데이터기준일자', '관광객 유형'], axis=1)

X['강수량'] = np.nan
X['평균기온'] = np.nan
X['최고기온'] = np.nan
X['최저기온'] = np.nan

X.head()
```

- ✓ 예측에 영향이 없는 관광객 유형, 데이터 기준일자 column 제거
- ✓ 강수량, 평균기온, 최고기온, 최저기온 column 추가

기준년월	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용금액	카드이용건수	건당이용금액	강수량	평균기온	최고기온	최저기온
2014-09	서귀포시	대륜동	농축수산품	여	50대	14434000	67	215433	NaN	NaN	NaN	NaN
2014-09	서귀포시	대륜동	농축수산품	남	50대	15119000	72	209986	NaN	NaN	NaN	NaN
2014-09	서귀포시	대륜동	농축수산품	여	40대	7609500	41	185598	NaN	NaN	NaN	NaN
2014-09	서귀포시	남원읍	농축수산품	남	50대	7092500	40	177313	NaN	NaN	NaN	NaN
2014-09	서귀포시	대륜동	농축수산품	남	40대	9098500	59	154212	NaN	NaN	NaN	NaN

데이터 병합

날씨 data – 서귀포시, 성산읍, 제주시

```
tmp = pd.read_csv('서귀포.csv', encoding='cp949')

for i in range(13146):
  if X.iloc[i, :]['제주 대분류'] == '서귀포시':
    for j in range(24):
      if X.iloc[i, :]['기준년월'] == tmp.iloc[j, :]['기준년월']:
        X['강수량'][i] = tmp['강수량'][j]
        X['평균기온'][i] = tmp['평균기온'][j]
        X['최저기온'][i] = tmp['최저기온'][j]
        X['최고기온'][i] = tmp['최고기온'][j]
```

- ✓ 관광객의 제주 대/중분류와 기준년월에 맞는 강수량, 평균기온, 최저기온, 최고기온 값 넣어줌
- ✓ 이후 행들을 섞어줌

```
1 X = X.sample(frac=1)
2 X.head()
```

	기준년월	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용금액	카드이용건수	건당이용금액	강수량	평균기온	최고기온	최저기온
5271	2015-07	서귀포시	성산읍	농축수산품	남	40대	11173300	187	59750	381.5	23.9	32.9	15.4
11755	2016-06	제주시	애월읍	약국	여	50대	342300	21	16300	162.7	22.2	30.2	16.7
8067	2015-12	제주시	연동	정장(여성)	여	20대	4166700	58	71840	90.0	10.0	20.1	2.3
10343	2016-04	서귀포시	남원읍	농축수산품	남	20대	1074300	25	42972	232.2	15.8	22.7	10.8
1223	2014-11	제주시	노형동	기타음료식품	남	40대	1501140	50	30023	100.3	13.9	22.5	6.9

데이터 병합

```
X = X.drop(['카드이용금액'], axis=1)
```

```
XX = X.reindex(columns=['기준년월', '강수량','평균기온','최고기온','최저기온', '제주 대분류', '제주 중분류', '업종명', '성별', '연령대별', '카드이용건수', '건당이용금액'])
```

```
XX.head()
```

	기준년월	강수량	평균기온	최고기온	최저기온	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용건수	건당이용금액
5271	2015-07	381.5	23.9	32.9	15.4	서귀포시	성산읍	농축수산품	남	40대	187	59750
11755	2016-06	162.7	22.2	30.2	16.7	제주시	애월읍	약국	여	50대	21	16300
8067	2015-12	90.0	10.0	20.1	2.3	제주시	연동	정장(여성)	여	20대	58	71840
10343	2016-04	232.2	15.8	22.7	10.8	서귀포시	남원읍	농축수산품	남	20대	25	42972
1223	2014-11	100.3	13.9	22.5	6.9	제주시	노형동	기타음료식품	남	40대	50	30023

```
from sklearn.model_selection import train_test_split
X_train, X_test = train_test_split(XX, test_size=0.2, random_state=42)
```

```
X_test.head()
```

	기준년월	강수량	평균기온	최고기온	최저기온	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용건수	건당이용금액
1420	2014-11	96.1	15.0	24.9	7.4	서귀포시	남원읍	슈퍼 마켓	남	30대	114	13455
404	2014-09	178.7	24.3	29.8	16.8	서귀포시	남원읍	스넥	여	30대	322	12499
8931	2016-01	125.6	6.1	16.8	-5.8	제주시	삼도2동	스넥	여	40대	29	14362
12094	2016-07	203.2	25.8	31.5	20.7	서귀포시	남원읍	기념품 점	여	50대	11	41818
8355	2015-12	90.0	10.0	20.1	2.3	제주시	연동	약국	남	50대	133	15591

- ✓ [카드이용금액 = 카드이용건수 x 건당이용금액] 이므로, 카드이용금액 column은 drop하고 건당이용금액 column만 남겨 label로 사용
- ✓ 8:2의 비율로 train과 test 데이터셋 나눔

Baseline code

데이터 전처리

```
X['연도'] = X['기준년월'].str.split('-').str[0].astype(int)
X['월'] = X['기준년월'].str.split('-').str[1].astype(int)
X = X.drop([ '기준년월', '건당이용금액'], axis=1)

test['연도'] = test['기준년월'].str.split('-').str[0].astype(int)
test['월'] = test['기준년월'].str.split('-').str[1].astype(int)
test = test.drop(['기준년월'], axis=1)
```

```
list =['제주 대분류', '제주 중분류', '업종명', '성별', '연령대별']

from sklearn import preprocessing
le = preprocessing.LabelEncoder()

for c in list:
    label = pd.concat([X[c], test[c]], axis=0)
    le.fit(label)
    X[c] = le.transform(X[c])
    test[c] = le.transform(test[c])
```

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler
sc = MinMaxScaler()
X_sc = sc.fit_transform(X)
test_sc = sc.transform(test)
```

	기준년월	강수량	평균기온	최고기온	최저기온	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용건수	건당이용금액
0	2015-10	61.0	18.9	25.5	10.7	서귀포시	예래동	슈퍼 마켓	여	20대	30	10881
1	2015-09	172.9	23.2	28.6	18.4	제주시	연동	약국	여	30대	239	12597
2	2015-11	215.5	15.4	22.6	2.1	서귀포시	남원읍	슈퍼 마켓	여	40대	131	21265
3	2016-04	232.2	15.8	22.7	10.8	서귀포시	예래동	기념품 점	여	40대	29	17000
4	2016-04	121.4	15.6	24.5	9.8	제주시	연동	슈퍼 마켓	여	50대	369	22050

- ✓ 기준년월 column은 연도와 월이 묶여 있기 때문에,
  '-' 을 기준으로 연도와 월을 나누어 각각의 column으로 만들어줌
- ✓ 문자 데이터들은 LabelEncoder를 통해 정수형 데이터로 맵핑
- ✓ MinmaxScaler를 통한 데이터 정규화

Baseline code

모델 설계

```
# DNN 모델 만들기
linear1 = torch.nn.Linear(12, 256)
linear2 = torch.nn.Linear(256, 128)
linear3 = torch.nn.Linear(128, 128)
linear4 = torch.nn.Linear(128, 1)
relu = torch.nn.ReLU() # relu 사용
dropout = torch.nn.Dropout(0.2)

# weight 초기화
torch.nn.init.xavier_normal_(linear1.weight)
torch.nn.init.xavier_normal_(linear2.weight)
torch.nn.init.xavier_normal_(linear3.weight)
torch.nn.init.xavier_normal_(linear4.weight)

model = torch.nn.Sequential(linear1, relu, dropout,
                            linear2,  relu, dropout,
                            linear3,  relu, dropout,
                            linear4).to(device)
```

- ✓ Layer를 4개 쌓아 DNN 모델을 만들고, 활성함수로는 relu 사용
- ✓ 과적합을 막기위해 0.2의 dropout값 설정
- ✓ Xavier 방식으로 weight 초기화

모델 학습

```
model.train()
loss = torch.nn.L1Loss()
optimizer = optim.Adam(model.parameters(), lr=1e-2)

for epoch in range(10000):
    hypothesis = model(X_sc)
    cost = loss(hypothesis, y.unsqueeze(1))
    optimizer.zero_grad()
    cost.backward()
    optimizer.step()
    if epoch%1000==0:
        print(epoch, cost.item())
```

```
0 35100.23828125
1000 17965.34765625
2000 14471.115234375
3000 9598.62890625
4000 9040.890625
5000 8580.6455078125
6000 8350.7939453125
7000 7962.37060546875
8000 7775.63037109375
9000 7432.36376953125
```

- ✓ Loss 함수는 MAE방식으로 cost를 측정하는 L1Loss 사용
- ✓ Adam optimizer를 사용해 0.01의 lr로 10000번의 epoch학습

Defence code

데이터 전처리

	강수량	평균기온	최고기온	최저기온	제주 대분류	제주 중분류	업종명	성별	연령대별	카드이용건수
count	10516.000000	10516.000000	10516.000000	10516.000000	10516.000000	10516.000000	10516.000000	10516.000000	10516.000000	10516.00000
mean	144.535470	16.901883	25.847651	9.261316	0.662800	4.627615	6.634557	0.501712	1.485926	132.430202
std	83.672119	6.972154	5.999427	8.190967	0.472776	2.860087	3.458367	0.500021	1.088081	210.386765
min	31.000000	5.000000	14.700000	-6.900000	0.000000	0.000000	0.000000	0.000000	0.000000	10.000000
25%	90.000000	10.400000	20.500000	1.100000	0.000000	2.000000	4.000000	0.000000	1.000000	23.000000
50%	121.400000	18.800000	27.500000	10.800000	1.000000	5.000000	6.000000	1.000000	1.000000	54.000000
75%	183.500000	22.800000	30.200000	16.800000	1.000000	7.000000	11.000000	1.000000	2.000000	142.250000
max	426.500000	28.500000	36.700000	22.300000	1.000000	9.000000	13.000000	1.000000	3.000000	2574.000000

```
q1, q3 = np.percentile(X['카드이용건수'], [25, 75])
iqr = q3 - q1
lower_bound = q1 - (iqr * 1.5)
upper_bound = q3 + (iqr * 1.5)
X['카드이용건수'][X['카드이용건수']<lower_bound] = lower_bound
X['카드이용건수'][X['카드이용건수']>upper_bound] = upper_bound
test['카드이용건수'][test['카드이용건수']<lower_bound] = lower_bound
test['카드이용건수'][test['카드이용건수']>upper_bound] = upper_bound
```

- ✓ Column들의 데이터 분포를 살펴본 결과, '카드이용건수' max값이 평균적인 데이터 분포에 비해 상당히 큰 것을 볼 수 있음
- ✓ 해당 column에 이상치가 존재한다고 판단하여 IQR값을 이용해 이상치를 처리해줌
- ✓ 이상치가 있는 행들을 버리게 되면 data수가 많이 줄기 때문에 각 bound 값으로 설정

Defense code

모델 설계

```
# DNN 모델 만들기
linear1 = torch.nn.Linear(X.shape[1], 512)
linear2 = torch.nn.Linear(512, 256)
linear3 = torch.nn.Linear(256, 128)
linear4 = torch.nn.Linear(128, 1)
relu = torch.nn.LeakyReLU()
dropout = torch.nn.Dropout(0.2)

# weight 초기화
torch.nn.init.kaiming_normal_(linear1.weight)
torch.nn.init.kaiming_normal_(linear2.weight)
torch.nn.init.kaiming_normal_(linear3.weight)
torch.nn.init.kaiming_normal_(linear4.weight)

model = torch.nn.Sequential(linear1, relu, dropout,
                            linear2, relu, dropout,
                            linear3, relu, dropout,
                            linear4).to(device)
```

- ✓ 히든 레이어수 늘림
- ✓ 활성함수 leakyrelu 로 변경
- ✓ kaiming 방식으로 weight 초기화

모델 학습

```
model.train()
loss = torch.nn.L1Loss()
optimizer = optim.Adam(model.parameters(), lr=1e-2)

for epoch in range(2500):
    hypothesis = model(X_sc)
    cost = loss(hypothesis, y.unsqueeze(1))
    optimizer.zero_grad()
    cost.backward()
    optimizer.step()
    if epoch%100==0:
        print(epoch, cost.item())
```

- ✓ 히든 레이어의 수를 늘렸기 때문에, epoch 수는 줄임
- ✓ 어느정도 epoch 이상 학습시, 과적합되어 오히려 accuracy 가 떨어짐 → cost가 처음 수렴하여 진동하기 시작하는 2500 epoch까지만 학습