Predict Future Sales

- 대회 링크 : https://www.kaggle.com/c/competitive-data-science-predict-future-sales/overview (https://www.kaggle.com/c/competitive-data-science-predict-future-sales/overview)
- 대회 개요 : 러시아의 최대 소프트웨어 회사인 1C Company에서 제공하는 일상적인 영업 데이터.
- 대회 문제 : 다음달에 모든 제품과 가게의 총 매출을 예상해 줄 것에 대한 요청
- 평가 방법 : RMSE(root mean squaed error)
- 제출 형식 : 데이터 셋의 각 ID에 대해 총 판매수를 예측하기

```
ID, item_cnt_month
0, 0.5
1, 0.5
3, 0.5
```

In [ ]:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

데이터 불러오기

In [ ]:

```
train = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/sales_train.csv")
test = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/test.csv")
sub = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/sample_submission.csv
items = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/items.csv")
items_cat = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/item_categories
shops = pd.read_csv("/kaggle/input/competitive-data-science-predict-future-sales/shops.csv")
```

- sales_train.csv : 학습 데이터. 2013년 1월부터 2015년 10월까지의 일일 기록 데이터.
- test.csv - 테스트 데이터. 상점과 제품의 2015년 11월 매출을 예측.
- sample_submission.csv : 제출용 샘플 파일
- items.csv : 항목/제품에 대한 추가 정보
- item_categories.csv : 항목 카테고리에 대한 추가 정보
- shops.csv : 상점에 대한 추가 정보

파일명	내용	행열
sales_train.csv	학습 데이터. 2013년 1월부터 2015년 10월까지의 일일 기록 데이터	2935849행, 6열
test.csv	테스트 데이터. 상점과 제품의 2015년 11월 매출을 예측	214200행, 3열
items.csv	항목/제품에 대한 추가 정보	22170행, 3열

파일명	내용	행열
item_categories.csv	항목 카테고리에 대한 추가 정보	84행, 2열
shops.csv	상점에 대한 추가 정보	60행, 2열
sample_submission.csv	올바른 형식의 샘플 파일 제출	

기본 데이터 탐색

In [ ]:

```
print("학습용 데이터 행열 : {}".format(train.shape))
print("제출용 데이터 행열 : {}".format(sub.shape))
print("테스트 데이터 행열 : {}".format(test.shape))
print("items 데이터 행열 : {}".format(items.shape))
print("items_categories 데이터 행열 : {}".format(items_cat.shape))
print("shops 데이터 행열 : {}".format(shops.shape))
```

데이터 살펴보기 (head())

In [ ]:

```
train.head(3)
```

In [ ]:

```
test.head(3)
```

In [ ]:

```
sub.head(3)
```

In [ ]:

```
items.head(3)
```

In [ ]:

```
items_cat.head(3)
```

In [ ]:

```
shops.head(3)
```

컬럼명 확인

In [ ]:

```
print("₩n 학습용 데이터 : {}".format(train.columns))
print("₩n 제출용 데이터 : {}".format(sub.columns))
print("₩n 테스트 데이터 : {}".format(test.columns))
print("₩n items 데이터 : {}".format(items.columns))
print("₩n items_categories 데이터 : {}".format(items_cat.columns))
print("₩n shops 데이터 : {}".format(shops.columns))
```

데이터 필드 설명

구분	컬럼명	설명	값
train	date	dd / mm / yyyy 형식의 날짜	날짜 데이터
train	date_block_num	편의를 위해 사용되는 연속 월 번호입니다.	2013/01(1)~2015/10(33)
train	shop_id	상점 고유 ID	0~59
train	item_id	항목 ID	0~22169
train	item_price	상품의 현재 가격	-1~307980
train	item_cnt_day	판매 된 제품 수입니다. 이 측정 값의 월별 금액을 예측하고 있습니다.	-22~2169
test	ID	테스트 예측을 위한 ID	0~214199
test	shop_id	상점 고유 ID	2~59
test	item_id	항목 ID	30~22167
sub	ID	테스트 예측을 위한 ID	0~214199
sub	item_cnt_month	예측해야 하는 값	default:0.5
items	item_name	항목 이름	범주의 개수(22170) '! ВО ВЛАСТИ НАВАЖДЕНИЯ (ПЛАСТ.) D', '!ABBYY FineReader 12 Professional Edition Full [PC, Цифровая версия]'
items	item_id	항목 ID	0~22169
items	item_category_id	항목 카테고리의 고유 식별자	0~83
items_categories	item_category_name	항목 카테고리 이름	범주의 개수(84) 'PC - Гарнитуры/Наушники' 'Аксессуары - PS2' 'Аксессуары - PS3' 'Аксессуары - PS4'
items_categories	item_category_id	항목 카테고리의 고유 식별자	0~83
shops	shop_name	상점 이름	범주의 개수(60)
shops	shop_id	상점 고유 ID	0~59

item_cnt_day 컬럼

- 판매 된 제품 수입니다.

In [ ]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:

```
plt.hist(train['item_cnt_day'])
```

In [ ]:

```
train.describe()
```

In [ ]:

```
test.describe()
```

In [ ]:

```
sub.describe()
```

In [ ]:

```
print(items.describe())
print(items_cat.describe())
print(shops.describe())
```

총 카테고리 개수

In [ ]:

```
def col_cat_col(col_name):
    num = len(col_name.unique() )
    print("범주의 개수 : {}".format(num) )
    print("List : ", col_name.unique() )
```

In [ ]:

```
col_cat_col(shops.shop_name)
```

In [ ]:

```
col_cat_col(items_cat.item_category_name)
```

In [ ]:

```
col_cat_col(items.item_name)
```

날짜 확인 후, 열 생성

In [ ]:

```
train['date'] = pd.to_datetime(train['date'],format = '%d.%m.%Y')
train['year'] = train['date'].dt.year
train['month'] = train['date'].dt.month
```

In [ ]:

```
# item_cnt_day : 판매된 제품수, item_price : 총 합계 금액
sum_train = train.groupby( ['year', 'month'] ).sum()
sum_train = sum_train.drop(['date_block_num', 'shop_id', 'item_id'], axis=1)
sum_train
```

sum_train과 원본 train 데이터합치기

In [ ]:

```
train.head()
```

In [ ]:

```
sum_train.head()
```

In [ ]:

```
test.head()
```

year, month를 기준 열로 삼아, 두 데이터 셋을 합친다.

In [ ]:

```
train_df = pd.merge(left=train,
                    right=sum_train,
                    how='left', on=['year', 'month'], sort=False)
train_df
```

모델 선택, 학습 및 예측

In [ ]:

```
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
```

입력, 출력 열 선택

In [ ]:

```
# 변수 선택 및 데이터 지정
sel = ['shop_id', 'item_id']
X_tr_all = train_df[sel]
X_test_all = test[sel]
```

In [ ]:

```
label = "item_cnt_day_y"
y_tr_all = train_df[label]
```

In [ ]:

```
X_train, X_test, y_train, y_test = train_test_split(X_tr_all, y_tr_all,
                                                    random_state=77)
```

In [ ]:

```
model = LinearRegression()  # 모델 생성
model.fit(X_train, y_train)  # 모델 훈련

model.score(X_test, y_test)
```

In [ ]:

```
# %%time

# model = RandomForestRegressor()  # 모델 생성
# model.fit(X_train, y_train)  # 모델 훈련

# model.score(X_test, y_test)
```

In [ ]:

```
model = LinearRegression()  # 모델 생성
model.fit(X_train, y_train)  # 모델 훈련
pred = model.predict(X_test_all) # 모델로 예측

sub['item_cnt_month'] = pred
sub
```

제출

In [ ]:

```
sub.to_csv("firstSub.csv", index=False)
```