캐글 대회 입문하기

대회 개요

- 대회명 : 미래 판매 예측
  - '데이터 과학 대회에서 우승하는 방법' Coursera 과정의 최종 프로젝트 대회
- 참여팀 : 2020/09/06 현재 8562개팀
- 데이터 : 러시아 최대 소프트웨어 회사 중 하나 인 1C Company 에서 친절하게 제공 한 일일 판매 데이터
- 우리의 과제 : 다음달의 모든 제품 및 매장에 대한 총 매출을 예측하기
- 평가지표 : 평균 제곱근 오차(RMSE)

제공된 데이터 위치 확인

In [2]:

```
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))
```

```
/kaggle/input/competitive-data-science-predict-future-sales/items.csv
/kaggle/input/competitive-data-science-predict-future-sales/sample_submission.csv
/kaggle/input/competitive-data-science-predict-future-sales/item_categories.csv
/kaggle/input/competitive-data-science-predict-future-sales/sales_train.csv
/kaggle/input/competitive-data-science-predict-future-sales/shops.csv
/kaggle/input/competitive-data-science-predict-future-sales/test.csv
```

파일 설명

- 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 : 올바른 형식의 샘플 파일 제출
- items.csv : 항목/제품에 대한 추가 정보
- item_categories.csv : 항목 카테고리에 대한 추가 정보
- shops.csv : 상점에 대한 추가 정보

In [3]:

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

기본 데이터 탐색

In [4]:

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

```
학습용 데이터 행열 : (2935849, 6)
제출용 데이터 행열 : (214200, 2)
테스트 데이터 행열 : (214200, 3)
```

In [5]:

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2935849 entries, 0 to 2935848
Data columns (total 6 columns):
 #   Column          Dtype  
---  ------          -----  
 0   date            object 
 1   date_block_num  int64  
 2   shop_id         int64  
 3   item_id         int64  
 4   item_price      float64
 5   item_cnt_day    float64
dtypes: float64(2), int64(3), object(1)
memory usage: 134.4+ MB
```

In [6]:

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214200 entries, 0 to 214199
Data columns (total 3 columns):
 #   Column   Non-Null Count   Dtype
---  ------   --------------   -----
 0   ID       214200 non-null  int64
 1   shop_id  214200 non-null  int64
 2   item_id  214200 non-null  int64
dtypes: int64(3)
memory usage: 4.9 MB
```

In [7]:

```
sub.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 214200 entries, 0 to 214199
Data columns (total 2 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   ID              214200 non-null  int64
 1   item_cnt_month  214200 non-null  float64
dtypes: float64(1), int64(1)
memory usage: 3.3 MB
```

데이터 필드 설명

- ID- 테스트 세트 내에서 (Shop, Item) 튜플을 나타내는 Id
- shop_id- 상점의 고유 식별자
- item_id- 상품의 고유 식별자
- item_category_id- 항목 카테고리의 고유 식별자
- item_cnt_day- 판매 된 제품 수입니다. 이 측정 값의 월별 금액을 예측하고 있습니다.
- item_price- 상품의 현재 가격
- date -dd / mm / yyyy 형식의 날짜
- date_block_num- 편의를 위해 사용되는 연속 월 번호입니다. 2013 년 1 월은 0, 2013 년 2 월은 1, ..., 2015 년 10 월은 33입니다.
- item_name- 항목 이름
- shop_name- 상점 이름
- item_category_name- 항목 카테고리 이름

In [8]:

```
train.head()
```

Out[8]:

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day
0	02.01.2013	0	59	22154	999.00	1.0
1	03.01.2013	0	25	2552	899.00	1.0
2	05.01.2013	0	25	2552	899.00	-1.0
3	06.01.2013	0	25	2554	1709.05	1.0
4	15.01.2013	0	25	2555	1099.00	1.0

In [9]:

```
test.head()
```

Out[9]:

	ID	shop_id	item_id
0	0	5	5037
1	1	5	5320
2	2	5	5233
3	3	5	5232
4	4	5	5268

In [10]:

```
print(sub.head())
print(sub.item_cnt_month.describe())
```

```
   ID  item_cnt_month
0   0             0.5
1   1             0.5
2   2             0.5
3   3             0.5
4   4             0.5
count    214200.0
mean          0.5
std           0.0
min           0.5
25%           0.5
50%           0.5
75%           0.5
max           0.5
Name: item_cnt_month, dtype: float64
```

In [11]:

```
print(train['item_cnt_day'].describe())
```

```
count    2.935849e+06
mean     1.242641e+00
std      2.618834e+00
min     -2.200000e+01
25%      1.000000e+00
50%      1.000000e+00
75%      1.000000e+00
max      2.169000e+03
Name: item_cnt_day, dtype: float64
```

In [37]:

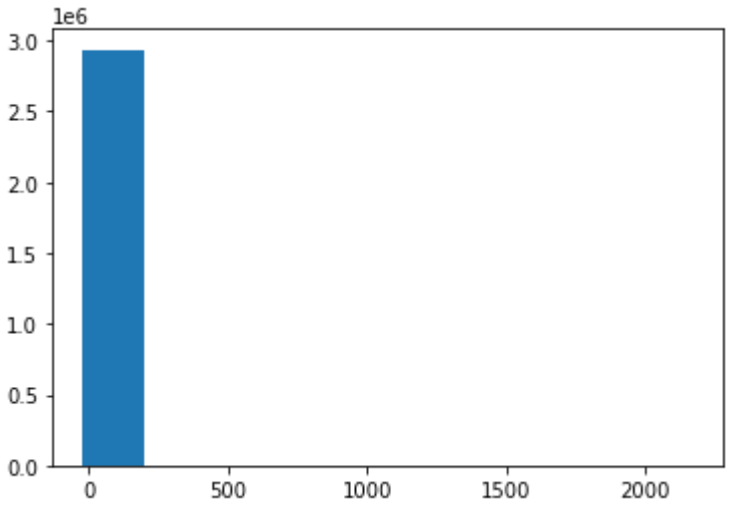
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
```

In [38]:

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

Out[38]:

```
(array([2.93581e+06, 2.40000e+01, 1.10000e+01, 2.00000e+00, 1.00000e+00,
        0.00000e+00, 0.00000e+00, 0.00000e+00, 0.00000e+00, 1.00000e+00]),
 array([ -22. ,  197.1,  416.2,  635.3,  854.4, 1073.5, 1292.6, 1511.7,
        1730.8, 1949.9, 2169. ]),
 <a list of 10 Patch objects>)
```

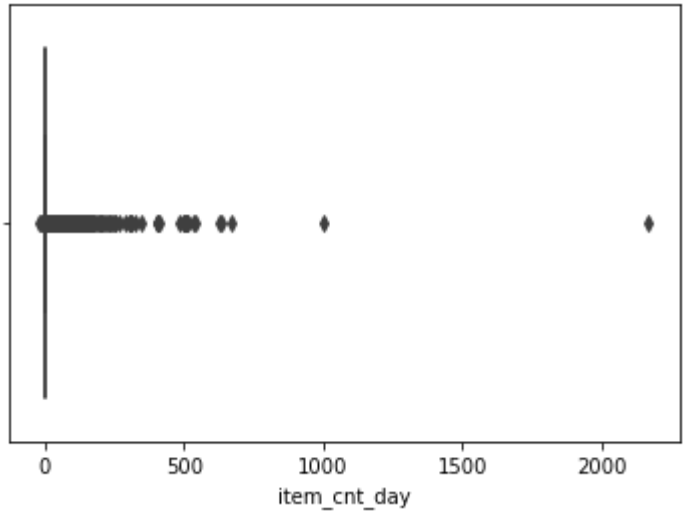


In [39]:

```
sns.boxplot(train['item_cnt_day'])
```

Out[39]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f1876fd82d0>
```



In [40]:

```
### 날짜 변환
train['date'] = pd.to_datetime(train['date'],format = '%d.%m.%Y')
```

In [41]:

```
train['year'] = train['date'].dt.year
train['month'] = train['date'].dt.month
```

In [42]:

```
train
```

Out[42]:

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day	year	month
0	2013-01-02	0	59	22154	999.00	1.0	2013	1
1	2013-01-03	0	25	2552	899.00	1.0	2013	1
2	2013-01-05	0	25	2552	899.00	-1.0	2013	1
3	2013-01-06	0	25	2554	1709.05	1.0	2013	1
4	2013-01-15	0	25	2555	1099.00	1.0	2013	1
...	...	...	...	...	...	...	...	...
2935844	2015-10-10	33	25	7409	299.00	1.0	2015	10
2935845	2015-10-09	33	25	7460	299.00	1.0	2015	10
2935846	2015-10-14	33	25	7459	349.00	1.0	2015	10
2935847	2015-10-22	33	25	7440	299.00	1.0	2015	10
2935848	2015-10-03	33	25	7460	299.00	1.0	2015	10

2935849 rows × 8 columns

In [43]:

```
sum_train = train.groupby( ['year', 'month'] ).sum()
sum_train = sum_train.drop(['date_block_num', 'shop_id', 'item_id', 'item_price'], axis=1)
sum_train
```

Out[43]:

		item_cnt_day
year	**month**	
2013	**1**	131479.0
	2	128090.0
	3	147142.0
	4	107190.0
	5	106970.0
	6	125381.0
	7	116966.0
	8	125291.0
	9	133332.0
	10	127541.0
	11	130009.0
	12	183342.0
2014	**1**	116899.0
	2	109687.0
	3	115297.0
	4	96556.0
	5	97790.0
	6	97429.0
	7	91280.0
	8	102721.0
	9	99208.0
	10	107422.0
	11	117845.0
	12	168755.0
2015	**1**	110971.0
	2	84198.0
	3	82014.0
	4	77827.0
	5	72295.0
	6	64114.0
	7	63187.0

		item_cnt_day
year	month	
	8	66079.0
	9	72843.0
	10	71056.0

In [44]:

```
sum_train.columns = ['all_item_cnt_month']
sum_train.columns
```

Out[44]:

```
Index(['all_item_cnt_month'], dtype='object')
```

In [45]:

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

Out[45]:

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day	year	month	all_
0	2013-01-02	0	59	22154	999.00	1.0	2013	1	
1	2013-01-03	0	25	2552	899.00	1.0	2013	1	
2	2013-01-05	0	25	2552	899.00	-1.0	2013	1	
3	2013-01-06	0	25	2554	1709.05	1.0	2013	1	
4	2013-01-15	0	25	2555	1099.00	1.0	2013	1	
...	...	...	...	...	...	...	...	...	
2935844	2015-10-10	33	25	7409	299.00	1.0	2015	10	
2935845	2015-10-09	33	25	7460	299.00	1.0	2015	10	
2935846	2015-10-14	33	25	7459	349.00	1.0	2015	10	
2935847	2015-10-22	33	25	7440	299.00	1.0	2015	10	
2935848	2015-10-03	33	25	7460	299.00	1.0	2015	10	

2935849 rows × 9 columns

In [46]:

```
test.head()
```

Out[46]:

	ID	shop_id	item_id
0	0	5	5037
1	1	5	5320
2	2	5	5233
3	3	5	5232
4	4	5	5268

In [47]:

```
# 변수 선택 및 데이터 지정
sel_f = ['shop_id', 'item_id']
X_train = train_df[sel_f]
X_test = test[sel_f]
```

In [48]:

```
label = "all_item_cnt_month"
y_train = train_df[label]
```

모델 만들기

In [49]:

```
from sklearn.linear_model import LinearRegression
```

In [50]:

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

Out[50]:

```
array([115018.02904017, 114996.78432021, 115003.31538253, ...,
       113826.66240028, 113534.56626825, 114936.79285528])
```

In [51]:

```
sub.columns
```

Out[51]:

```
Index(['ID', 'item_cnt_month'], dtype='object')
```

In [52]:

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

Out[52]:

	ID	item_cnt_month
0	0	115018.029040
1	1	114996.784320
2	2	115003.315383
3	3	115003.390452
4	4	115000.687944
...	...	...
214195	214195	113624.199468
214196	214196	113794.307367
214197	214197	113826.662400
214198	214198	113534.566268
214199	214199	114936.792855

214200 rows × 2 columns

제출

In [53]:

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

두번째 모델

shop_id, item_id 별 월별 합계

In [54]:

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

Out[54]:

				item_price	item_cnt_day
year	month	shop_id	item_id		
			32	884.0	6.0
			33	1041.0	3.0
2013	1	0	35	247.0	1.0
			43	221.0	1.0
			51	257.0	2.0
...	...	...	...	...	...
			22087	357.0	6.0
			22088	238.0	2.0
2015	10	59	22091	179.0	1.0
			22100	629.0	1.0
			22102	1250.0	1.0

1609124 rows × 2 columns

In [55]:

```
sum_train.columns = ['shop_item_price_month', 'shop_item_cnt_month']
sum_train.columns
```

Out[55]:

```
Index(['shop_item_price_month', 'shop_item_cnt_month'], dtype='object')
```

In [56]:

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

Out[56]:

	date	date_block_num	shop_id	item_id	item_price	item_cnt_day	year	month	shc
0	2013-01-02	0	59	22154	999.00	1.0	2013	1	
1	2013-01-03	0	25	2552	899.00	1.0	2013	1	
2	2013-01-05	0	25	2552	899.00	-1.0	2013	1	
3	2013-01-06	0	25	2554	1709.05	1.0	2013	1	
4	2013-01-15	0	25	2555	1099.00	1.0	2013	1	
...	...	...	...	...	...	...	...	...	...
2935844	2015-10-10	33	25	7409	299.00	1.0	2015	10	
2935845	2015-10-09	33	25	7460	299.00	1.0	2015	10	
2935846	2015-10-14	33	25	7459	349.00	1.0	2015	10	
2935847	2015-10-22	33	25	7440	299.00	1.0	2015	10	
2935848	2015-10-03	33	25	7460	299.00	1.0	2015	10	

2935849 rows × 10 columns

In [57]:

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

label = "shop_item_cnt_month"
y_tr_all = train_df[label]

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

Out[57]:

```
0.004599368506762125
```

In [58]:

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

model.score(X_test, y_test)
```

Out[58]:

```
0.004599368506762125
```

In [59]:

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

In [ ]:

```

```