Алгоритам за динамичко поставување на цени на производи од различни категории врз база на побарувачката и конкуренцијата

(https://www.kaggle.com/datasets/anirudhchauhan/retail-store-inventory-forecasting-dataset)

Филип Костов,

191115

GitHub: https://github.com/FilipKostov/DSproject

1. Вовед

Целта на овој проект е да направиме алгоритам за динамичко поставување на цени на производи врз податоци, подредени во временска серија, за производите како нивна побарувачка во даден ден, конкурентииска цена на производите, попусти и промоции, цена на претходни продажби, залиха, број на продадени единки од производи во даден ден, категории на производи, локација на производи (во која продавница се наоѓаат). Предвидување на цени на производи, а и во глобала на какви и да е асети, дали цени на берза или криптовалути, е прилично тешка работа поради тоа што треба да се енкапсулираат надворешни трендови кои што носат огромна информација и треба да се пристапи со добро селектирана техника на feature engineering за извлекување на податоци со скриени значења .

	Date	Store ID	Product ID	Category	Region	Inventory Level	Units Sold
1	2022-01-01	S001	P0001	Groceries	North	231	
2	2022-01-01	S001	P0002	Toys	South	204	
3	2022-01-01	S001	P0003	Toys	West	102	
4	2022-01-01	S001	P0004	Toys	North	469	
5	2022-01-01	S001	P0005	Electronics	East	166	
6	2022-01-01	S001	P0006	Groceries	South	138	
7	2022-01-01	S001	P0007	Furniture	East	359	
8	2022-01-01	S001	P0008	Clothing	North	380	
9	2022-01-01	S001	P0009	Electronics	West	183	
10	2022-01-01	S001	P0010	Toys	South	108	
11	2022-01-01	S001	P0011	Furniture	South	258	
12	2022-01-01	S001	P0012	Clothing	West	66	
13	2022-01-01	S001	P0013	Toys	South	96	

1. Retail data set

Клучни цели на проектот се:

- анализа на податоците и нивно претпроцесирање
- feature engineering – релевантно на предвидувањето на цени
- тренирање на ML модел за предвидување на цени
- евалуирање на перформанси на модел
-

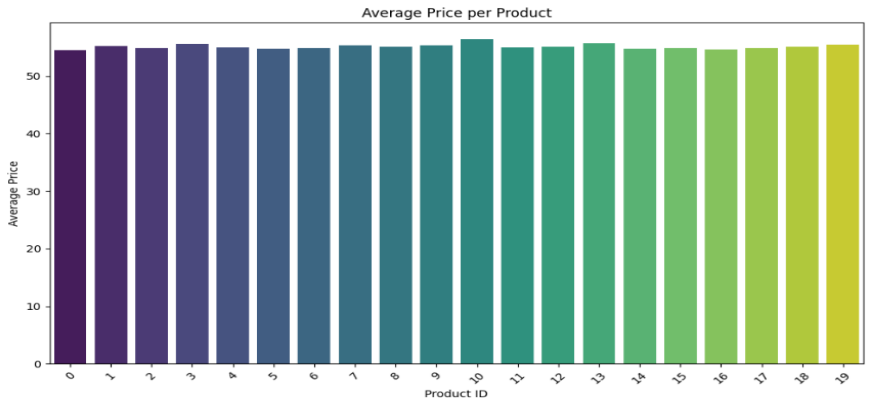
2. Анализа на податоците и визуелизација

Податочното множество е прилично големо, со 73100 набљудувања. Доменот на категориите податоци во податочното множество е прилично мал, со само 20 врсти на производи кои припаѓаат на 5 категории, во 5 различни продавници.

```
Date: 731 unique categories
Store ID: 5 unique categories
Product ID: 20 unique categories
Category: 5 unique categories
Region: 4 unique categories
Weather Condition: 4 unique categories
Seasonality: 4 unique categories
```

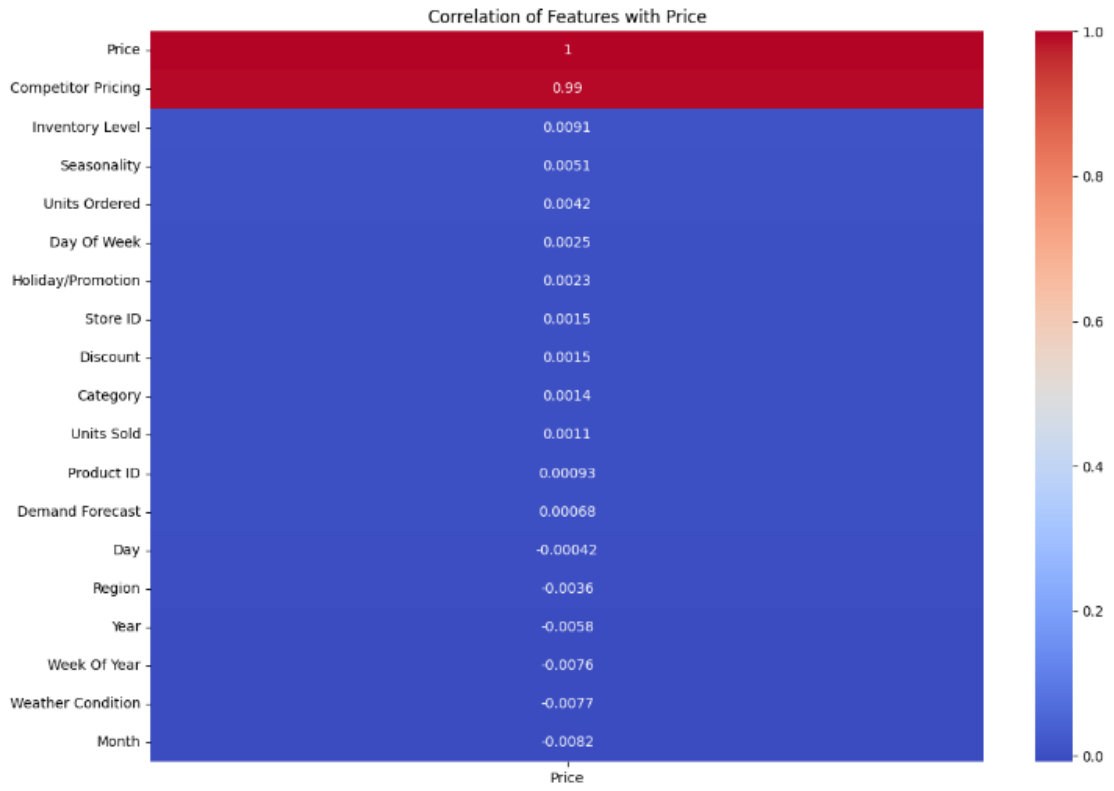
2. Categorical Data Domain

Средната цена на секој различен производ е приближна до 55.



3. Product price mean

Преку матрица на корелација на податоците, забележав дека конкурентската цена има строго линеарна корелација со цената на производите.



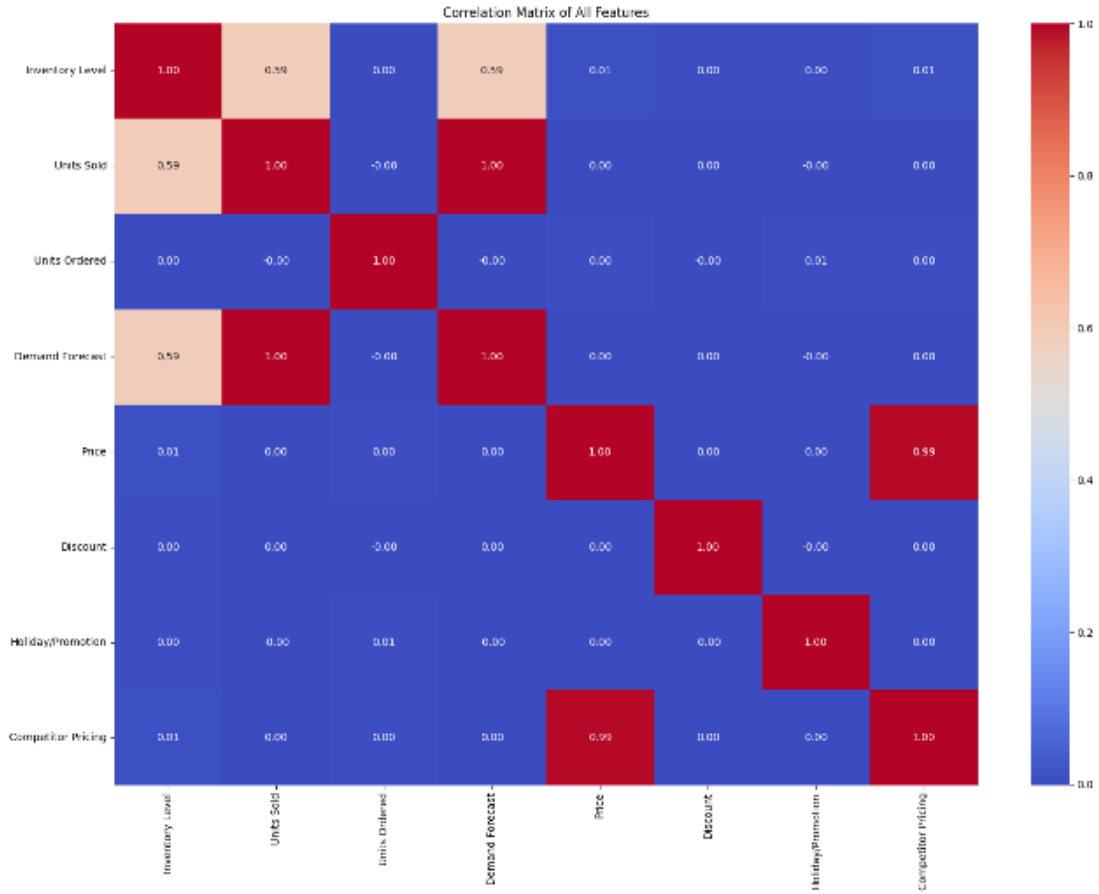
4. Correlation matrix for Price

На сликата од матрицата на корелација може да се забележи дека освен конкурентска цена на производи, другите нумерички карактеристики на податоците имаат многу ниска, скоро никаква линеарна корелација со цената на производите. Ова е чудно затоа што преку емпирииски обзервации може да се заклучи дека карактеристики како продадени единки, предвидената побарувачка итн. би имале посилна корелација со цената на производите. За проверка, освен Пирсонова матрица на корелација, ја измерив корелацијата на цената со карактеристиката продадени единки со Спирман корелација. Резултатот е следен:

```
Spearman correlation: 5.65400562322617 24e-05
```

5. Spearman correlation

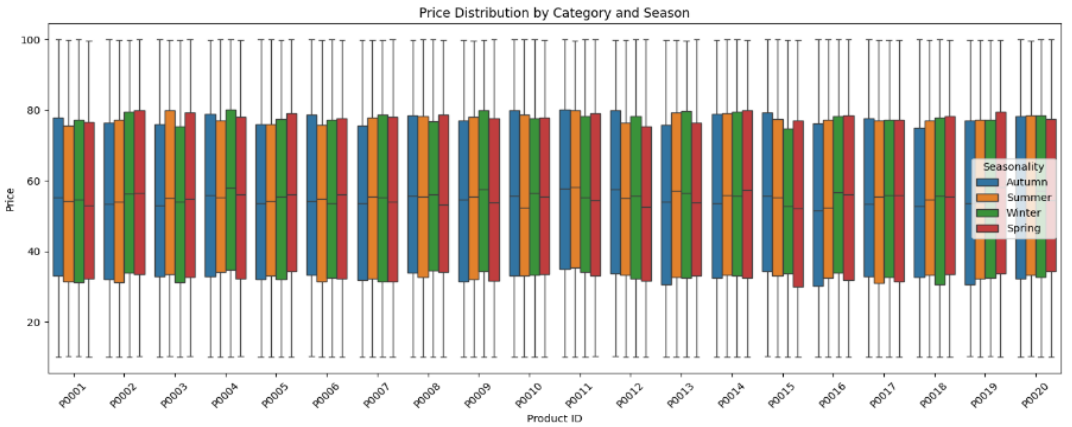
Ова мерење значи дека пак нема никаква корелација помеѓу цената и продадените единки. Преку матрица на линеарна корелација на секоја со секоја карактеристика, наведена на сликата подолу, се гледа дека некои карактеристики имаат високи корелации со други, имено предвидената побарувачка има висока корелација од 1.0 со бројот на продадени единки за даден производ, како и корелација од 0.59 со бројот на производи во инвентар.



6. Correlation Matrix

Бројот на продадени единки исто така има висока корелација со бројот на производи во инвентар, што и при емпириска обзервација може да се забележи дека е така.

Цената на производите има корелација со категоријата “Seasonality”, и ова се гледа преку box-plot-овите на сликата подолу. Просечната цена на даден производ е поголема во неговата наведена сезона.



6. Seasonality Box plots

3. Предпроцесирање на податоците

При проверка, податочното множество нема податоци кои недостасуваат, null вредности, па затоа не е потребна техника за справување со тоа.

```
Date                    0
Store ID                0
Product ID              0
Category                0
Region                  0
Inventory Level         0
Units Sold              0
Units Ordered           0
Demand Forecast         0
Price                   0
Discount                0
Weather Condition       0
Holiday/Promotion       0
Competitor Pricing      0
Seasonality             0
```

7. Check for null data

Во податочното множество има податоци од категориски тип во неколку од карактеристиките. Бидејќи одлучив да користам XGBoost модел како најоптимален, истите категориски податоци ги енкодирав користејќи Label енкодер, а затоа што користам XGBoost модел, кој е граден на дрва на одлука, не е потребно да ги нормализирам податоците во одреден ранг, затоа што за дрвата на одлука не е потребна информацијата за рангот и магнитудата на податоците, туку нивниот релативен редослед, всушност и тренирање на дрво на одлука на нормализирани податоци е лоша техника затоа што при нормализирање се губат информации и интерпретабилност.

```
enc_cat_data=["Store ID", "Product ID", "Category", "Region", "Weather Condition", "Seasonality"]

label_encoders=dict()
for cat in enc_cat_data:
    lb=LabelEncoder()
    retailDf[cat]=lb.fit_transform(retailDf[cat])
    label_encoders[cat]=lb
```

8. Label encoding categorical data

4. Податочното множество како временска серија

Податочното множество претставува временска серија. Секој влез содржи карактеристика која носи информации за датум (во формат година, месец, ден), па затоа, бидејќи XGBoost моделот неможе да работи врз податоци од типот datetime, пристапот кој го превземав е да ги извлечам информациите за ден, месец и година поединечно од datetime податоците, како и додадов дополнителни карактеристики во податочното множество за која недела во годината и ден во неделата. При проверка на перформансите на моделот, еднаш трениран со овие карактеристики, еднаш без нив, моделот извади понизок Root Mean Square Error и повисок R^2 скор кога беше трениран со овие карактеристики. Ова е заради тоа што карактеристиката Seasonality има корелација со цената на производите, како и датумот што има корелација со сезоната на податоците, па овие карактеристики кои што ги додадов на податочното множество му носат информации на моделот.

```
RMSE: 2.83
MAE: 2.42
R² Score: 0.9882
```

9. Performance metrics

Затоа што податочното множество претставува временска серија, преку feature engineering потребно е да воведеме карактеристики кои ќе носат информации за временски зависности помеѓу обзервациите. Одлучив

за секоја обзервација да додадам 7 карактеристики кои носат информација за бројот на продадени единки во последните 7 денови. Ова го направив преку функција која ја зима вредноста од карактеристиката Units Sold од дадена обзервација и ја поставува како вредност во една од тие 7 карактеристики за продажбата на производите во последните 7 денови, при првично групирање на податоците според Product ID и Store ID, затоа што податоците иницијално беа подредени според datetime објектот, како временска серија, и не групирани. Одбрав овој lag да биде 7 обзервации (7 денови) затоа што најоптимално ја енкапсулира информацијата, затоа што податочното множество содржи карактеристики за дали дадениот ден е викенд и дали има промоција, како и сезоната за секој производ. Бидејќи претходно увидовме дека сезоната на производите има корелација со нивната цена, сетирање на висок lag, да речеме од 90, не ги енкапсулира скриените неделни трендови на цената на производите, како и тоа како тие се менуваат за време на викендски денови. Истото го направив за предвидената побарувачка и конкуренциската цена. Преку овој пристап, бидејќи обзервациите ги групирав по Product Id и Store Id, некои од обзервациит, поточно првите 7 обзервации од секоја група ќе имаат податоци што недостасуваат, и бидејќи податочното множество ни е доволно големо (73100), одлучив овие обзервации да ги изфрлам. Бидејќи е временска серија овој пристап носи до поголема загуба на информации одколку податочно множество кое не е влијателно на време, затоа што понатаму податочното множество кога го поделувам на тестирачко и тренирачко, го делам на секвенцијален начин, а не на интерплетен начин, а оваа носи информација. Сликата подолу претставува пример како се губат овие податоци поради користење на lag и шифтање на истите.

	Product ID	Store ID	Date	Units Sold	Prev_Units
0	1	1	2025-01-01	10	NaN
1	1	1	2025-01-02	12	10.0
2	1	2	2025-01-01	8	NaN
3	2	1	2025-01-01	20	NaN
4	2	1	2025-01-02	25	20.0

10. Example of NaN in introducing lag

Исто така за секоја додадив карактеристика за средната вредност на продадени единки, побарувачка и конкуренциска цена од последните 7 обзервации со .rolling(window=7).

При поделба на податоците на тренирачко и тестирачко множество, наидов на концепт кој што претставува проблем во науката за податоците, одпрвин неможе лесно да се забележи, а тоа е data leakage. Со воведување на lag, некои од податоците од тренирачкото множество влеваат во податоците на тестирачкото множество. Тоа е затоа што го делиме множеството секвенцијално, точно 80% од првите податоци според нивниот датум, од 2022-2024 влеваат во тренирачкото множество, првите 7 обзервации од тестирачкото множество на овој начи ќе содржат податоци кои припаѓаат на тренирачкото множество. Бидејќи овие податоци се од претходни (според време) обзервации, и не се идни податоци, ова не претставува проблем за нашето податочно множество, па одлучив да го земам пристапот врз целото податочно множество да воведам lag, а потоа да го поделам.

5. Тренирање на модел XGBoost

Затоа што податочното множество енкапсулира трендови помеѓу карактеристиките, кои се не линеарни и имаат високо-димензионална зависност, како и фактот што податочното множество содржи и нумерички и категориски податоци, како и врските помеѓу податоците според нивниот датум, одбрав да користам XGBoost модел. Бидејќи XGBoost не се справува со податоци од типот datetime, како и тоа што веќе ги извлековме поединечно податоците за ден, месец, година, како нумерички вредности, ја извадив карактеристиката “Date” и во тренирачкото и во тестирачкото множество. Таргет варијабла, која што моделот е потребно да ја предвиди е цената. При fine-tunning на моделот забележав дека поголема длабочина на нивоата на дрвата кои ги користи XGBoost води кон полоши перформанси и најдобри перформанси вади при 6 нивоа, воочени на сликите подолу:

```
RMSE: 2.90
MAE: 2.47
R² Score: 0.9876
```

Оваа се резултатите од длабочина од 10 нивоа

```
RMSE: 2.84
MAE: 2.42
R² Score: 0.9881
```

Ова се резултатите од длабочина од 6 нивоа

При поголем број на дрва (500:) и при тренирање на секое дрво со помал број на семплови (60% од обзервациите за тренирање на секое дрво) моделот вади подобри перформанси одколку помал број на дрва кои се тренирани со поголем број на обзервации (n-estimators=100, subsamples=0.8).

```
RMSE: 2.84
MAE: 2.42
R² Score: 0.9882
```

Ова се резултатите од тренирање на моделот со n-estimators=500, subsamples=0.6

Може да се забележи дека, доколку минимален, моделот вади подобар R^2 скор.

6. Заклучок

Во овој проект дизајнирав модел за предвидување на цени на производи, базиран на регресија користејќи XGBoost. Со feature-engineering на временски карактеристики како воведувањето на lag за карактеристиките Units Sold, Demand Forecast и Competitor Price, успешно ги опфатив временските трендови на податочното множество.

При fine-tunning на моделот, добив пожелни резултати преку користење на овие метрики:

```python
model = XGBRegressor(
    n_estimators=500,
    learning_rate=0.05,
    max_depth=6,
    subsample=0.6,
    colsample_bytree=0.8,
    random_state=42
)
```

што укажуваат на силна предикативна моќ и добра генерализација на податоците. Анализата на важноста на карактеристиките потврди дека претходните трендови на продажба, идентитетот на производот, цените на конкурентите и сезонските фактори носат клучни информации за оптимизација на цените.

Резултатите покажуваат дека стратегиите за поставување на цени, базирани на податоци, можат ефикасно да се се реализираат преку науката за податоци и машинското учење, овозможувајќи им на бизнисите оптимизирана профитабилност низ различни производи и услуги.