Practica1-RegressionGPA404

October 12, 2022

1 Apartat (C): Analitzant Dades

[1]:

```
from sklearn.datasets import make_regression
import numpy as np
import pandas as pd
%matplotlib inline
from matplotlib import pyplot as plt
import scipy.stats
import seaborn as sns;
import warnings
warnings.filterwarnings("ignore", category=DeprecationWarning)

# Visualitzarem només 3 decimals per mostra
pd.set_option('display.float_format', lambda x: '%.3f' % x)

# Funcio per a llegir dades en format csv
def load_dataset(path):
    dataset = pd.read_csv(path, header=0, delimiter=',')
    return dataset

# Carreguem dataset d'exemple
dataset = load_dataset('data/OnionPrices.csv')
# Eliminem la columna commodity perque es irrelevant
del dataset["commodity"]

dataset.drop(dataset.loc[dataset['max_price'] == 0].index, inplace=True)
dataset.drop(dataset.loc[dataset['min_price'] == 0].index, inplace=True)

#Guardem les noves dades en un csv i les carguem un altre cop
dataset.to_csv("data/OnionPrices_2.csv")
dataset = load_dataset('data/OnionPrices_2.csv')
del dataset["Unnamed: 0"]

data = dataset.values
x = data[:, :]
y_modal_price = data[:, -1]
```

[2]:

```
# Visualitzar les dades
dataset.head()
```

[2]:

```
             state district   market variety arrival_date  min_price  \
0  Andhra Pradesh  Kurnool  Kurnool   Local   03/01/2020       1350
1  Andhra Pradesh  Kurnool  Kurnool   Local   04/01/2020       1390
2  Andhra Pradesh  Kurnool  Kurnool   Local   06/01/2020       1460
3  Andhra Pradesh  Kurnool  Kurnool   Local   07/01/2020       2010
4  Andhra Pradesh  Kurnool  Kurnool   Local   10/01/2020       1320

   max_price  modal_price
0       4390     3100.000
1       4400     3200.000
2       5150     4310.000
3       5200     4200.000
4       4050     3300.000
```

[3]:

```
# Comprova quants valors de cada atribut son únics
print("Numero de mercados:", len(dataset['market'].unique()))
print("Numero de distritos:", len(dataset['district'].unique()))
print("Numero de estados:", len(dataset['state'].unique()))
print("Numero de variedad:", len(dataset['variety'].unique()))
```

```
Numero de mercados: 905
Numero de distritos: 315
Numero de estados: 22
Numero de variedad: 21
```

[5]:

```
#Mostrem de cada atribut categoritzable els seus valors únics.
print("Estados:", dataset['state'].unique())
print("=================================")
print("Variedad:", dataset['variety'].unique())
print("=================================")
print("Market:", dataset['market'].unique())
print("=================================")
print("District:", dataset['district'].unique())
```

```
Estados: ['Andhra Pradesh' 'Chattisgarh' 'Goa' 'Gujarat' 'Haryana'
 'Himachal Pradesh' 'Jammu and Kashmir' 'Jharkhand' 'Karnataka' 'Kerala'
 'Madhya Pradesh' 'Maharashtra' 'Nagaland' 'NCT of Delhi' 'Odisha'
 'Punjab' 'Rajasthan' 'Telangana' 'Tripura' 'Uttar Pradesh' 'Uttrakhand'
 'West Bengal']
=================================
Variedad: ['Local' 'Other' 'Onion' 'Nasik' 'Red' 'White' 'Beelary-Red' '1st
Sort'
 'Bangalore-Samall' 'Puna' 'Pusa-Red' 'Bombay (U.P.)' 'Telagi' 'Hybrid'
 'Big' 'Small' '2nd Sort' 'Pole' 'Dry F.A.Q.' 'Medium' 'Bellary']
=================================
Market: ['Kurnool' 'Pattikonda' 'Tiphra' 'Durg' 'Raigarh' 'Rajnandgaon' 'Mapusa'
 'Ahmedabad(Chimanbhai Patal Market Vasana)' 'Dhari'
 'Anand(Veg,Yard,Anand)' 'Khambhat(Veg Yard Khambhat)'
 'Petlad(Veg Yard, Petlad)' 'Deesa(Deesa Veg Yard)' 'Bharuch' 'Bhavnagar'
 'Mahuva(Station Road)' 'Dahod(Veg. Market)' 'Jamnagar' 'Visavadar'
 'Kapadvanj' 'Nadiyad(Piplag)' 'Mehsana(Mehsana Veg)' 'Bilimora' 'Godhra'
 'Porbandar' 'Gondal' 'Jetpur(Dist.Rajkot)' 'Morbi' 'Rajkot(Ghee Peeth)'
 'Songadh' 'Surat' 'Vadhvan' 'Padra' 'Vadodara(Sayajipura)'
 'Ambala Cantt.' 'Ambala City' 'Barara' 'Mullana' 'Naraingarh'
 'Shahzadpur' 'Ch. Dadri' 'Ballabhgarh' 'Faridabad' 'Fatehabad' 'Jakhal'
 'Farukh Nagar' 'Gurgaon' 'Pataudi' 'Sohna' 'Barwala(Hisar)' 'Hansi'
 'Narnaund' 'Uklana' 'Bahadurgarh' 'Jhajjar' 'Jind' 'Narwana' 'Safidon'
 'Dhand' 'Pundri' 'Siwan' 'Asandh' 'Gharaunda'
 'New Grain Market(main), Karnal' 'Babain' 'Iamailabad' 'Ladwa' 'Pehowa'
 'Pipli' 'Shahabad' 'Thanesar' 'Mohindergarh' 'Nuh' 'Punhana' 'Taura'
 'Hassanpur' 'Palwal' 'Barwala' 'New Grain Market , Panchkula'
 'Panchkul(Kalka)' 'Madlauda' 'Panipat' 'Samalkha' 'Rewari' 'Meham'
 'New Grain Market , Rohtak' 'Sampla' 'Ellanabad' 'kalanwali' 'Rania'
 'Sirsa' 'Ganaur' 'Gohana' 'Sonepat' 'Sonepat(Kharkhoda)' 'Chhachrauli'
 'Jagadhri' 'Mustafabad' 'Radaur' 'Sadhaura' 'Yamuna Nagar' 'Bilaspur'
 'Chamba' 'Hamirpur' 'Hamirpur(Nadaun)' 'Kangra' 'Kangra(Baijnath)'
 'Kangra(Jaisinghpur)' 'Kangra(Jassour)' 'Kangra(Nagrota Bagwan)'
 'Palampur' 'Bhuntar' 'Kullu' 'Kullu(Chauri Bihal)' 'Dhanotu (Mandi)'
 'Mandi(Mandi)' 'Mandi(Takoli)' 'Rohroo' 'Shimla'
 'Shimla and Kinnaur(Rampur)' 'Nahan' 'Paonta Sahib' 'Solan'
 'Solan(Nalagarh)' 'Santoshgarh' 'Una' 'Ashahipora (Anantnagh)' 'Akhnoor'
 'Batote' 'Narwal Jammu (F&V)' 'Samba' 'Kathua' 'Rajouri (F&V)'
 'Parimpore' 'Reasi' 'Udhampur' 'Lohardaga' 'Khunti' 'Bagalakot'
 'Jamakhandi' 'Bangalore' 'Channapatana' 'Doddaballa Pur' 'Ramanagara'
 'Belgaum' 'Bellary' 'Hospet' 'Humanabad' 'Bijapur' 'Chamaraj Nagar'
 'Gundlupet' 'Bagepalli' 'Chikkamagalore' 'Kadur' 'Tarikere' 'Davangere'
 'Dharwar' 'Hubli (Amaragol)' 'Gadag' 'Arakalgud' 'Arasikere' 'Belur'
 'Channarayapatna' 'Hassan' 'Haveri' 'Ranebennur' 'Bangarpet'
 'Chickkaballapura' 'Chintamani' 'Gowribidanoor' 'Kolar' 'Malur'
 'Srinivasapur' 'K.R. Pet' 'Mandya' 'Mangalore' 'Hunsur'
 'Mysore (Bandipalya)' 'Nanjangud' 'T. Narasipura' 'Raichur' 'Bhadravathi'
 'Shimoga' 'Gubbi' 'Tiptur' 'Tumkur' 'Udupi' 'Alappuzha' 'Aroor'
 'Chengannur' 'Cherthala' 'Harippad' 'Kayamkulam' 'Madhavapuram' 'Mannar'
 'Aluva' 'Angamaly' 'Ernakulam' 'Kothamangalam' 'Moovattupuzha'
 'Perumbavoor' 'Piravam' 'Thrippunithura' 'Kattappana' 'Munnar'
 'Nedumkandam' 'Thodupuzha' 'Vandiperiyar' 'Kannur' 'Kanjangadu'
 'Neeleswaram' 'Anchal' 'Chathanoor' 'Athirampuzha' 'Ettumanoor'
 'Kanjirappally' 'Kottayam' 'Kuruppanthura' 'Pala' 'Pampady'
 'Thalayolaparambu' 'Kallachi' 'Mukkom' 'Palayam' 'Perambra' 'Quilandy'
 'Thamarassery' 'Kondotty' 'Kottakkal' 'Manjeri' 'Parappanangadi'
 'Perinthalmanna' 'Thirurrangadi' 'Koduvayoor' 'Palakkad' 'Pattambi'
 'vadakarapathy' 'Vadakkenchery' 'Chalakudy' 'Chavakkad' 'Chelakkara'
 'Irinjalakkuda' 'Kodungalloor' 'Thrissur' 'Wadakkanchery' 'Aralamoodu'
```

```
'Chala' 'Pothencode' 'Vamanapuram' 'Alirajpur(F&V)' 'Anuppur' 'Sendhwa'
'Bhopal(F&V)' 'Burhanpur(F&V)' 'Chhatarpur' 'Chhindwara(F&V)'
'Damoh(F&V)' 'Dewas(F&V)' 'Haatpipliya' 'Badnawar(F&V)' 'Dhamnod'
'Dhar(F&V)' 'Kukshi' 'Rajgarh' 'Guna(F&V)' 'Lashkar(F&V)' 'Harda(F&V)'
'Timarni' 'Hoshangabad' 'Pipariya' 'Gautampura' 'Indore(F&V)' 'Sanwer'
'Jabalpur(F&V)' 'Jhabua' 'Petlawad' 'Khandwa(F&V)' 'Pandhana(F&V)'
'Mandsaur' 'Shamgarh(F&V)' 'Sitmau' 'Morena(F&V)' 'Porsa(F&V)'
'Gadarwada' 'Javad' 'Manasa' 'Neemuch' 'Raisen' 'Biaora' 'Narsinghgarh'
'Ratlam(F&V)' 'Sailana' 'Deori' 'Sagar(F&V)' 'Satna(F&V)' 'Ashta'
'Sehore' 'Berachha' 'Kalapipal' 'Shajapur(F&V)' 'Shujalpur' 'Soyatkalan'
'Syopurkalan(F&V)' 'Shivpuri(F&V)' 'Ujjain(F&V)' 'Ahmednagar' 'Akole'
'Jamkhed' 'Kopargaon' 'Newasa(Ghodegaon)' 'Parner' 'Pathardi' 'Rahata'
'Rahuri' 'Rahuri(Vambori)' 'Sangamner' 'Shevgaon' 'Shrigonda'
'Shrirampur' 'Amarawati' 'Amrawati(Frui & Veg. Market)' 'Morshi'
'Aurangabad' 'Gangapur' 'Lasur Station' 'Paithan' 'Vaijpur' 'Kada'
'Malkapur' 'Nandura' 'Chandrapur(Ganjwad)' 'Dhule' 'Sakri' 'Shirpur'
'Chalisgaon' 'Jalgaon' 'Yawal' 'Kolhapur' 'Vashi New Mumbai' 'Kamthi'
'Nagpur' 'Ramtek' 'Navapur' 'Chandvad' 'Devala' 'Dindori' 'Dindori(Vani)'
'Kalvan' 'Lasalgaon' 'Lasalgaon(Niphad)' 'Lasalgaon(Vinchur)' 'Malegaon'
'Malegaon(Umarane)' 'Manmad' 'Nampur' 'Nandgaon' 'Nasik' 'Pimpalgaon'
'Pimpalgaon Baswant(Saykheda)' 'Satana' 'Sinner' 'Umrane' 'Yeola'
'Osmanabad' 'Washi(Thane  Market)' 'Dound' 'Indapur' 'Junnar'
'Junnar(Alephata)' 'Junnar(Otur)' 'Khed(Chakan)' 'Maanachar' 'Pune'
'Pune(Hadapsar)' 'Pune(Khadiki)' 'Pune(Manjri)' 'Pune(Moshi)'
'Pune(Pimpri)' 'Karjat(Raigad)' 'Ratnagiri (Nachane)'
'Sangli(Phale, Bhajipura Market)' 'Vita' 'Karad' 'Lonand' 'Palthan'
'Satara' 'Vai' 'Akluj' 'Barshi' 'Karmala' 'Kurdwadi' 'Kurdwadi(Modnimb)'
'Mangal Wedha' 'Pandharpur' 'Solapur' 'Kalyan' 'Murbad' 'Mokokchung Town'
'Azadpur' 'Keshopur' 'Shahdara' 'Angaura' 'Angul' 'Angul(Jarapada)'
'Talcher' 'Bampada' 'Barikpur' 'Jaleswar' 'Nilagiri' 'Attabira' 'Bargarh'
'Bargarh(Barapalli)' 'Godabhaga' 'Bhadrak' 'Chandabali' 'Sahidngar'
'Bolangir' 'Tusura' 'Boudh' 'Khunthabandha' 'Kendupatna'
'Kendupatna(Niali)' 'Dhenkanal' 'Hindol' 'Kamakhyanagar' 'Mottagaon'
'Bhanjanagar' 'Digapahandi' 'Hinjilicut' 'Jajpur' 'Jhumpura' 'Jharsuguda'
'Bhawanipatna' 'Junagarh' 'Kalahandi(Dharamagarh)' 'Kesinga'
'Chatta Krushak Bazar' 'Gopa' 'Kendrapara' 'Kendrapara(Marshaghai)'
'Pattamundai' 'Keonjhar' 'Keonjhar(Dhekikote)' 'Saharpada' 'Balugaon'
'Jatni' 'Malkanagiri' 'Malkangiri(Korakunda)' 'Baripada' 'Betnoti'
'Chuliaposi' 'Saraskana' 'Udala' 'Khariar' 'Khariar Road' 'Rayagada'
'Birmaharajpur' 'Dungurapalli' 'Pandkital' 'Bonai' 'Sargipali' 'Ajnala'
'Amritsar(Amritsar Mewa Mandi)' 'Gehri(Jandiala mandi)' 'Rayya' 'Barnala'
'Bathinda' 'Bhagta Bhai Ka' 'Bhucho' 'Goniana' 'Maur' 'Raman'
'Rampuraphul(Nabha Mandi)' 'Talwandi Sabo' 'Faridkot' 'Jaitu' 'Kotkapura'
'Bassi Pathana' 'Khamano' 'Sirhind' 'Abohar' 'Fazilka' 'Jalalabad'
'Ferozepur Cantt.' 'Firozepur City' 'Guru Har Sahai' 'Makhu' 'Mamdot'
'Zira' 'Batala' 'Dhariwal' 'Dinanagar' 'F.G.Churian' 'Gurdaspur'
'Quadian' 'Dasuya' 'Garh Shankar' 'Garh Shankar(Mahalpur)'
'GarhShankar (Kotfatuhi)' 'Hoshiarpur' 'Mukerian' 'Mukerian(Talwara)'
```

```
'Tanda Urmur' 'Adampur' 'Bhogpur' 'Bilga' 'Goraya' 'Nakodar'
'Phillaur(Apra Mandi)' 'Bhulath' 'Phagwara' 'Doraha' 'Jagraon' 'Khanna'
'Ludhiana' 'Machhiwara' 'Sahnewal' 'Samrala' 'Budalada' 'Mansa'
'Baghapurana' 'Dharamkot' 'Moga' 'Nihal Singh Wala' 'Banur'
'Banur (Kheragaju)' 'Dera Bassi' 'Kharar' 'Kurali' 'Lalru' 'Bariwala'
'Giddarbaha' 'Malout' 'Muktsar' 'Balachaur' 'Banga'
'Nawan Shahar(MandiRaho)' 'Nawan Shahar(Subzi Mandi)' 'Pathankot'
'Dudhansadhan' 'Ghanaur' 'Nabha' 'Patiala' 'Patran' 'Rajpura' 'Samana'
'Anandpur Sahib' 'Chamkaur Sahib' 'Morinda' 'Ropar' 'Bhawanigarh' 'Dhuri'
'Khanauri' 'Lehra Gaga' 'Malerkotla' 'Sangrur' 'Sunam' 'Patti'
'Tarantaran' 'Ajmer(F&V)' 'Beawar' 'Bijay Nagar' 'Kekri'
'Madanganj Kishanganj' 'Alwar(FV)' 'Khairthal' 'Balotra' 'Bayana'
'Nadwai' 'Bhilwara' 'Bikaner(F&V)' 'Chittorgarh' 'Nimbahera' 'Pratapgarh'
'Churu' 'Sujangarh(Churu)' 'Padampur' 'Sadulshahar' 'Sriganganagar(F&V)'
'Bhadara' 'Hanumangarh' 'Hanumangarh Town' 'Hanumangarh(Urlivas)'
'Pilli Banga' 'Sangriya' 'Suratgarh' 'Chomu(F&V)' 'Jaipur(Bassi)'
'Jaipur(F&V)' 'Jaisalmer' 'Jalore' 'Sanchor' 'Jhunjhunu' 'Nawalgarh'
'Jodhpur(F&V)(Bhadwasia)' 'Kota (FV)' 'Nagour(FV)' 'Sojat Road'
'Rajasamand' 'Sikar' 'Surajgarh' 'Abu Road' 'Tonk' 'Udaipur(F&V)'
'Bowenpally' 'Gudimalkapur' 'L B Nagar' 'Mahboob Manison' 'Sadasivpet'
'Gandacharra' 'Halahali' 'Kalyanpur' 'Teliamura' 'Dasda' 'Kadamtala'
'Kanchanpur' 'Panisagar' 'Bishalgarh' 'Bishramganj' 'Jumpuijala'
'Melaghar' 'Barpathari' 'Kalsi' 'Manubazar' 'Masmara' 'Achnera' 'Agra'
'Fatehpur Sikri' 'Jagnair' 'Jarar' 'Khairagarh' 'Samsabad' 'Aligarh'
'Atrauli' 'Charra' 'Khair' 'Ajuha' 'Allahabad' 'Jasra' 'Akbarpur' 'Tanda'
'Achalda' 'Auraiya' 'Dibiapur' 'Azamgarh' 'Badayoun' 'Bilsi' 'Shahaswan'
'Ujhani' 'Wazirganj' 'Bagpat' 'Baraut' 'Khekda' 'Bahraich' 'Naanpara'
'Risia' 'Ruperdeeha' 'Ballia' 'Chitwadagaon' 'Rasda' 'Vilthararoad'
'Balrampur' 'Tulsipur' 'Utraula' 'Atarra' 'Baberu' 'Banda' 'Barabanki'
'Rudauli' 'Safdarganj' 'Anwala' 'Bahedi' 'Bareilly' 'Basti' 'Gopiganj'
'Bijnaur' 'Chaandpur' 'Dhampur' 'Kiratpur' 'Nagina' 'Najibabad'
'Anoop Shahar' 'Buland Shahr' 'Divai' 'Gulavati' 'Jahangirabad' 'Khurja'
'Sikanderabad' 'Sikarpur' 'Siyana' 'Chandoli' 'Karvi' 'Barhaj' 'Devariya'
'Aliganj' 'Awagarh' 'Etah' 'Ganjdudwara' 'Kasganj' 'Bharthna' 'Etawah'
'Jasvantnagar' 'Faizabad' 'Farukhabad' 'Kamlaganj' 'Kayamganj'
'Mohamadabad' 'Bindki' 'Fatehpur' 'Jahanabad' 'Firozabad' 'Shikohabad'
'Sirsaganj' 'Tundla' 'Dadri' 'Dankaur' 'Ghaziabad' 'Hapur' 'Muradnagar'
'Noida' 'Gazipur' 'Jamanian' 'Jangipura' 'Saidpur' 'Yusufpur' 'Gonda'
'Karnailganj' 'Nawabganj' 'Chorichora' 'Gorakhpur' 'Sehjanwa'
'Bharuasumerpur' 'Maudaha' 'Muskara' 'Raath' 'Hardoi' 'Madhoganj' 'Sandi'
'Sandila' 'Shahabad(New Mandi)' 'Haathras' 'Shadabad' 'Ait' 'Jalaun'
'Konch' 'Orai' 'Jaunpur' 'Mugrabaadshahpur' 'Shahganj' 'Chirgaon'
'Gurusarai' 'Jhansi' 'Mauranipur' 'Moth' 'Amroha' 'Dhanura' 'Hasanpur'
'Chhibramau(Kannuj)' 'Kannauj' 'Choubepur' 'Jhijhank' 'Kanpur(Grain)'
'Pukhrayan' 'Rura' 'Uttaripura' 'Varipaal' 'Bharwari' 'Maigalganj'
'Mohammdi' 'Tikonia' 'Golagokarnath' 'Lakhimpur' 'Paliakala' 'Lalitpur'
'Lucknow' 'Anandnagar' 'Gadaura' 'Maharajganj' 'Nautnava' 'Partaval'
'Mahoba' 'Bewar' 'Ghiraur' 'Mainpuri' 'Kosikalan' 'Mathura' 'Kopaganj'
```

```
 'Mau' 'Mawana' 'Meerut' 'Sardhana' 'Mirzapur' 'Chandausi' 'Muradabad'
 'Sambhal' 'Kadhle' 'Kairana' 'Khatauli' 'Muzzafarnagar' 'Shahpur'
 'Shamli' 'Thanabhawan' 'Tamkuhi Road' 'Billsadda' 'Pilibhit' 'Puranpur'
 'Vishalpur' 'Bachranwa' 'Jayas' 'Lalganj' 'Raibareilly' 'Milak' 'Rampur'
 'Vilaspur' 'Chutmalpur' 'Devband' 'Gangoh' 'Nanuta' 'Rampurmaniharan'
 'Saharanpur' 'Sultanpurchilkana' 'Khalilabad' 'Badda' 'Katra' 'Puwaha'
 'Shahjahanpur' 'Tilhar' 'Naugarh' 'Sahiyapur' 'Soharatgarh' 'Wansi'
 'Hargaon (Laharpur)' 'Mehmoodabad' 'Sindholi' 'Sitapur' 'Viswan' 'Dudhi'
 'Robertsganj' 'Jafarganj' 'Sultanpur' 'Bangarmau' 'Purwa' 'Unnao'
 'Varanasi(F&V)' 'Tanakpur' 'Dehradoon' 'Rishikesh' 'Vikasnagar'
 'Kotadwara' 'Bhagwanpur(Naveen Mandi Sthal)' 'Haridwar Union' 'Lakshar'
 'Manglaur' 'Roorkee' 'Haldwani' 'Ramnagar' 'Jaspur(UC)' 'Kashipur'
 'Khateema' 'Kicchha' 'Rudrapur' 'Sitarganj' 'Bankura Sadar'
 'Bishnupur(Bankura)' 'Birbhum' 'Bolpur' 'Rampurhat' 'Sainthia' 'Asansol'
 'Burdwan' 'Durgapur' 'Kalna' 'Katwa' 'Dinhata' 'Mekhliganj'
 'Karsiyang(Matigara)' 'Siliguri' 'Sheoraphuly' 'Ramkrishanpur(Howrah)'
 'Uluberia' 'Alipurduar' 'Belacoba' 'Dhupguri' 'Falakata'
 'Jalpaiguri Sadar' 'Moynaguri' 'Bara Bazar (Posta Bazar)' 'English Bazar'
 'Gajol' 'Samsi' 'Egra/contai' 'Tamluk (Medinipur E)' 'Medinipur(West)'
 'Jangipur' 'Chakdah' 'Kalyani' 'Nadia' 'Ranaghat' 'Barasat' 'Habra'
 'Balarampur' 'Kasipur' 'Purulia' 'Baruipur(Canning)'
 'Diamond Harbour(South 24-pgs)']
==================================
District: ['Kurnool' 'Bilaspur' 'Durg' 'Raigarh' 'Rajnandgaon' 'North Goa'
 'Ahmedabad' 'Amreli' 'Anand' 'Banaskanth' 'Bharuch' 'Bhavnagar' 'Dahod'
 'Jamnagar' 'Junagarh' 'Kheda' 'Mehsana' 'Navsari' 'Panchmahals'
 'Porbandar' 'Rajkot' 'Surat' 'Surendranagar' 'Vadodara(Baroda)' 'Ambala'
 'Bhiwani' 'Faridabad' 'Fatehabad' 'Gurgaon' 'Hissar' 'Jhajar' 'Jind'
 'Kaithal' 'Karnal' 'Kurukshetra' 'Mahendragarh-Narnaul' 'Mewat' 'Palwal'
 'Panchkula' 'Panipat' 'Rewari' 'Rohtak' 'Sirsa' 'Sonipat' 'Yamuna Nagar'
 'Chamba' 'Hamirpur' 'Kangra' 'Kullu' 'Mandi' 'Shimla' 'Sirmore' 'Solan'
 'Una' 'Anantnag' 'Jammu' 'Kathua' 'Rajouri' 'Srinagar' 'Udhampur'
 'Lohardaga' 'Ranchi' 'Bagalkot' 'Bangalore' 'Belgaum' 'Bellary' 'Bidar'
 'Bijapur' 'Chamrajnagar' 'Chikmagalur' 'Davangere' 'Dharwad' 'Gadag'
 'Hassan' 'Haveri' 'Kolar' 'Mandya' 'Mangalore(Dakshin Kannad)' 'Mysore'
 'Raichur' 'Shimoga' 'Tumkur' 'Udupi' 'Alappuzha' 'Ernakulam' 'Idukki'
 'Kannur' 'Kasargod' 'Kollam' 'Kottayam' 'Kozhikode(Calicut)' 'Malappuram'
 'Palakad' 'Thirssur' 'Thiruvananthapuram' 'Alirajpur' 'Anupur' 'Badwani'
 'Bhopal' 'Burhanpur' 'Chhatarpur' 'Chhindwara' 'Damoh' 'Dewas' 'Dhar'
 'Guna' 'Gwalior' 'Harda' 'Hoshangabad' 'Indore' 'Jabalpur' 'Jhabua'
 'Khandwa' 'Mandsaur' 'Morena' 'Narsinghpur' 'Neemuch' 'Raisen' 'Rajgarh'
 'Ratlam' 'Sagar' 'Satna' 'Sehore' 'Shajapur' 'Sheopur' 'Shivpuri'
 'Ujjain' 'Ahmednagar' 'Amarawati' 'Aurangabad' 'Beed' 'Buldhana'
 'Chandrapur' 'Dhule' 'Jalgaon' 'Kolhapur' 'Mumbai' 'Nagpur' 'Nandurbar'
 'Nashik' 'Osmanabad' 'Pune' 'Raigad' 'Ratnagiri' 'Sangli' 'Satara'
 'Sholapur' 'Thane' 'Mokokchung' 'Delhi' 'Angul' 'Balasore' 'Bargarh'
 'Bhadrak' 'Bolangir' 'Boudh' 'Cuttack' 'Dhenkanal' 'Ganjam' 'Jajpur'
 'Jharsuguda' 'Kalahandi' 'Kendrapara' 'Keonjhar' 'Khurda' 'Malkangiri'
 'Mayurbhanja' 'Nuapada' 'Rayagada' 'Sonepur' 'Sundergarh' 'Amritsar'
 'Barnala' 'Bhatinda' 'Faridkot' 'Fatehgarh' 'Fazilka' 'Ferozpur'
 'Gurdaspur' 'Hoshiarpur' 'Jalandhar' 'kapurthala' 'Ludhiana' 'Mansa'
 'Moga' 'Mohali' 'Muktsar' 'Nawanshahr' 'Pathankot' 'Patiala'
 'Ropar (Rupnagar)' 'Sangrur' 'Tarntaran' 'Ajmer' 'Alwar' 'Barmer'
 'Bharatpur' 'Bhilwara' 'Bikaner' 'Chittorgarh' 'Churu' 'Ganganagar'
 'Hanumangarh' 'Jaipur' 'Jaisalmer' 'Jalore' 'Jhunjunu' 'Jodhpur' 'Kota'
 'Nagaur' 'Pali' 'Rajasamand' 'Sikar' 'Sirohi' 'Tonk' 'Udaipur'
 'Hyderabad' 'Medak' 'Dhalai' 'Khowai' 'North Tripura' 'Sepahijala'
 'South District' 'Unokoti' 'Agra' 'Aligarh' 'Allahabad' 'Ambedkarnagar'
 'Auraiya' 'Azamgarh' 'Badaun' 'Baghpat' 'Bahraich' 'Ballia' 'Balrampur'
 'Banda' 'Barabanki' 'Bareilly' 'Basti' 'Bhadohi(Sant Ravi Nagar)'
 'Bijnor' 'Bulandshahar' 'Chandauli' 'Chitrakut' 'Deoria' 'Etah' 'Etawah'
 'Faizabad' 'Farukhabad' 'Fatehpur' 'Firozabad' 'Gautam Budh Nagar'
 'Ghaziabad' 'Ghazipur' 'Gonda' 'Gorakhpur' 'Hardoi' 'Hathras'
 'Jalaun (Orai)' 'Jaunpur' 'Jhansi' 'Jyotiba Phule Nagar' 'Kannuj'
 'Kanpur' 'Kaushambi' 'Khiri (Lakhimpur)' 'Lakhimpur' 'Lalitpur' 'Lucknow'
 'Maharajganj' 'Mahoba' 'Mainpuri' 'Mathura' 'Mau(Maunathbhanjan)'
 'Meerut' 'Mirzapur' 'Muradabad' 'Muzaffarnagar' 'Padrauna(Kusinagar)'
 'Pillibhit' 'Pratapgarh' 'Raebarelli' 'Rampur' 'Saharanpur'
 'Sant Kabir Nagar' 'Shahjahanpur' 'Siddharth Nagar' 'Sitapur' 'Sonbhadra'
 'Sultanpur' 'Unnao' 'Varanasi' 'Champawat' 'Dehradoon' 'Garhwal (Pauri)'
 'Haridwar' 'Nanital' 'UdhamSinghNagar' 'Bankura' 'Birbhum' 'Burdwan'
 'Coochbehar' 'Darjeeling' 'Hooghly' 'Howrah' 'Jalpaiguri' 'Kolkata'
 'Malda' 'Medinipur(E)' 'Medinipur(W)' 'Murshidabad' 'Nadia'
 'North 24 Parganas' 'Puruliya' 'Sounth 24 Parganas']
```

[6]:

```
#Estadistiques sobre la nostre col·lecció de dades
print(dataset.describe())
```

```
       min_price  max_price  modal_price
count 107105.000 107105.000   107105.000
mean    1896.110   2293.933     2109.441
std     1459.381   1564.327     1493.184
min       20.000     54.000       20.000
25%     1000.000   1225.000     1150.000
50%     1400.000   1800.000     1600.000
75%     2400.000   2800.000     2550.000
max    18000.000  25000.000    22000.000
```

[7]:

```
# Desglosem el atribut arrival date, per mes i dia
dataset['arrival_date'] = pd.to_datetime(dataset['arrival_date'])
dataset['month'] = pd.DatetimeIndex(dataset['arrival_date']).month
dataset['day'] = pd.DatetimeIndex(dataset['arrival_date']).day

dataset
```

```
/tmp/ipykernel_18189/1990634617.py:2: UserWarning: Parsing dates in DD/MM/YYYY
```

```
format when dayfirst=False (the default) was specified. This may lead to
inconsistently parsed dates! Specify a format to ensure consistent parsing.
  dataset['arrival_date'] = pd.to_datetime(dataset['arrival_date'])
```

[7]:

```
                 state            district                         market  \
0       Andhra Pradesh             Kurnool                        Kurnool
1       Andhra Pradesh             Kurnool                        Kurnool
2       Andhra Pradesh             Kurnool                        Kurnool
3       Andhra Pradesh             Kurnool                        Kurnool
4       Andhra Pradesh             Kurnool                        Kurnool
…                    …                   …                              …
107100     West Bengal  Sounth 24 Parganas  Diamond Harbour(South 24-pgs)
107101     West Bengal  Sounth 24 Parganas  Diamond Harbour(South 24-pgs)
107102     West Bengal  Sounth 24 Parganas  Diamond Harbour(South 24-pgs)
107103     West Bengal  Sounth 24 Parganas  Diamond Harbour(South 24-pgs)
107104     West Bengal  Sounth 24 Parganas  Diamond Harbour(South 24-pgs)

       variety arrival_date  min_price  max_price  modal_price  month  day
0        Local   2020-03-01       1350       4390     3100.000      3    1
1        Local   2020-04-01       1390       4400     3200.000      4    1
2        Local   2020-06-01       1460       5150     4310.000      6    1
3        Local   2020-07-01       2010       5200     4200.000      7    1
4        Local   2020-10-01       1320       4050     3300.000     10    1
…          …           …         …          …          …          …   …
107100     Red   2020-03-09       2200       2300     2250.000      3    9
107101     Red   2020-04-09       2050       2600     2200.000      4    9
107102     Red   2020-08-09       2700       2875     2800.000      8    9
107103     Red   2020-09-09       2625       2875     2800.000      9    9
107104     Red   2020-10-09       2800       2890     2870.000     10    9

[107105 rows x 10 columns]
```

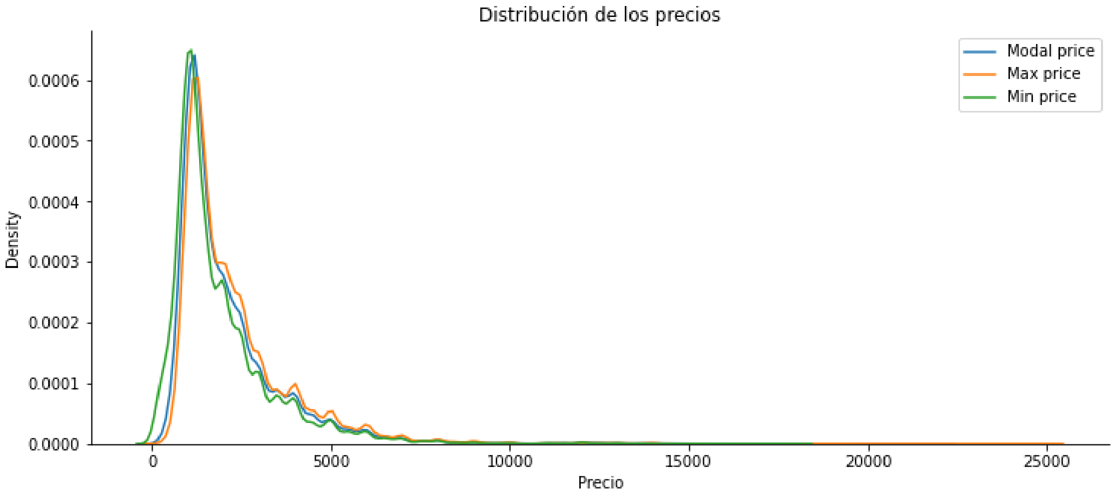
[8]:

```
# Es transforme l'atribut arrival_time que es un objecte al tipus data
print(dataset.dtypes)
```

```
state                   object
district                object
market                  object
variety                 object
arrival_date    datetime64[ns]
min_price                int64
max_price                int64
modal_price            float64
month                    int64
day                      int64
dtype: object
```

[313]:

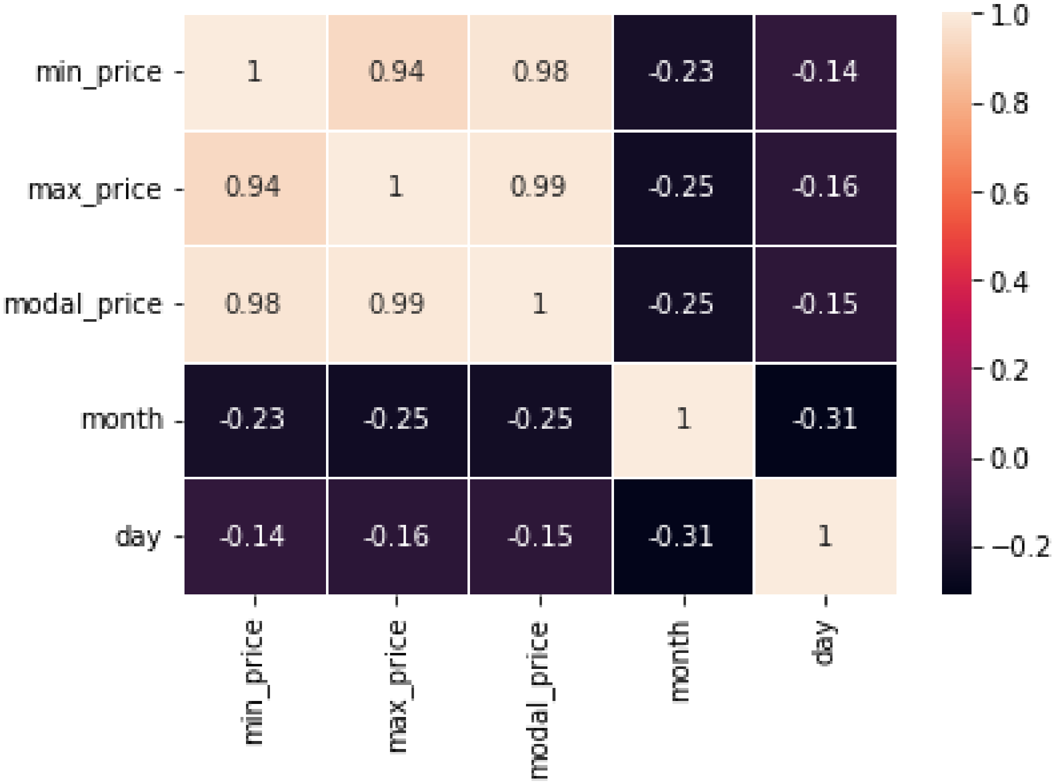
```python
#Gráfic sobre la densitat dels preus, comprovem quina distribució tenim.
plt.figure(figsize=(12, 5))
sns.kdeplot(data=dataset['modal_price'], label='Modal price')
sns.kdeplot(data=dataset['max_price'], label='Max price')
sns.kdeplot(data=dataset['min_price'], label='Min price' )

plt.title('Distribución de los precios')
plt.xlabel('Precio')
plt.legend()
sns.despine()
plt.show()
```



També podem estudiar la correlació entre els diferents atributs per tal de saber si estan correlacionats entre ells.

[314]:

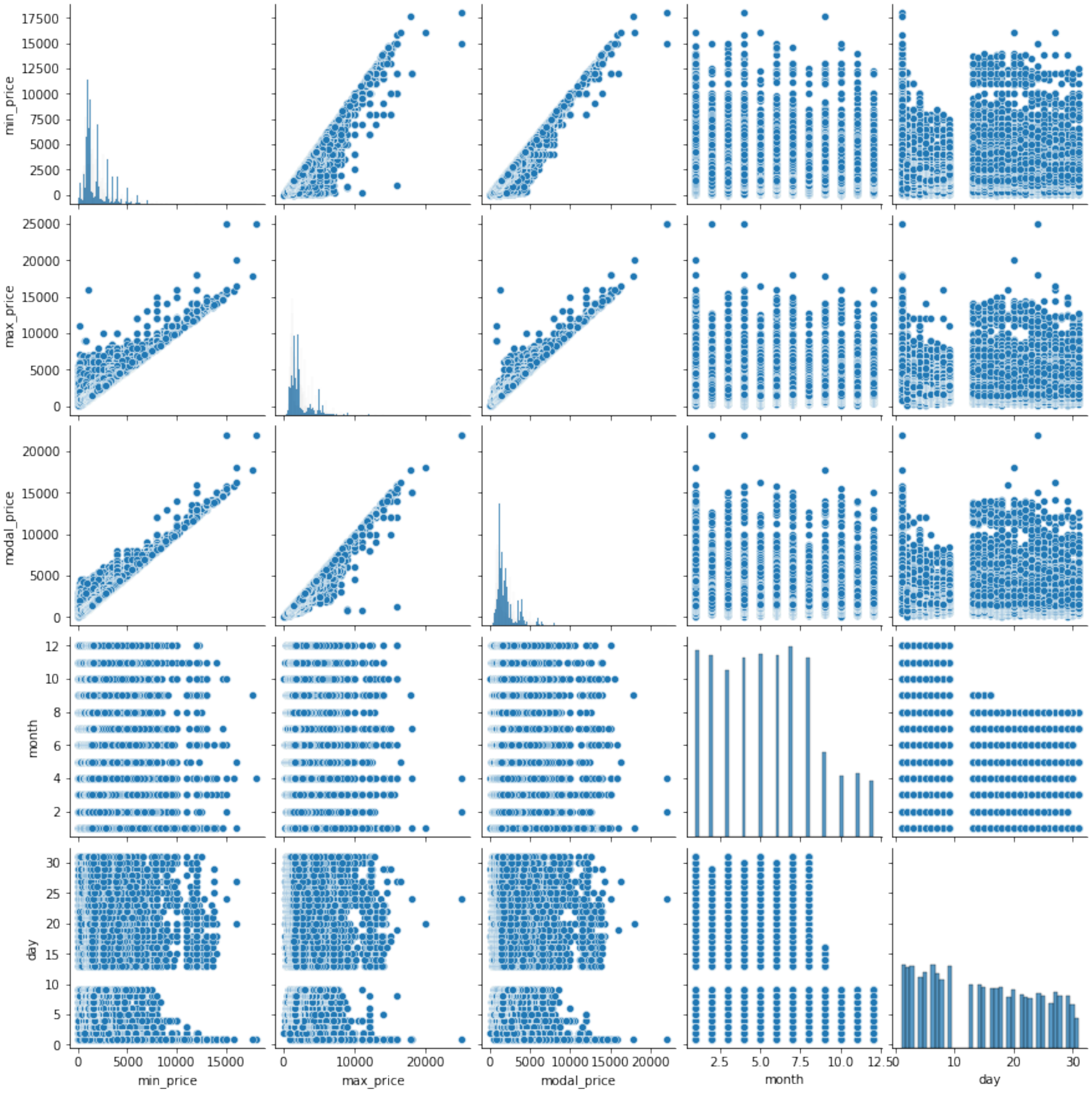
```python
import seaborn as sns

# Mirem la correlació entre els atributs d'entrada per entendre millor les dades
correlacio = dataset.corr()
plt.figure()
ax = sns.heatmap(correlacio, annot=True, linewidths=.5)
```



També podem utilitzar la funció pairplot per tal de veure els atributs que estan relacionats entre si.

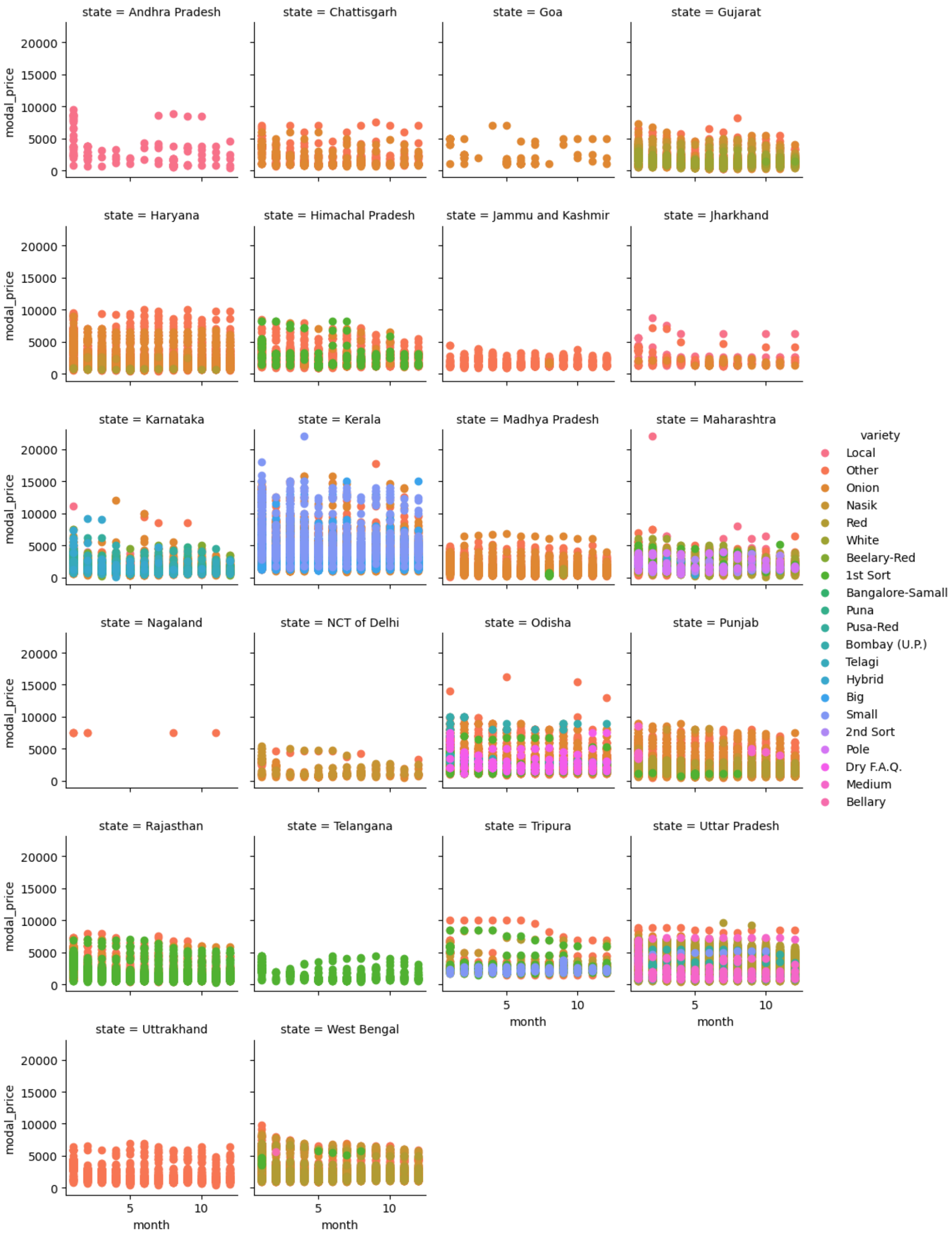
[315]:

```
# Mirem la relació entre atributs utilitzant la funció pairplot
relacio = sns.pairplot(dataset)
```



```
[20]: #Grafiques de cada estat sobre la varietat, on la x son els mesos, la y el preu␣
       ↪model.
      g = sns.FacetGrid(dataset,  col="state",hue="variety", col_wrap=4, height=2.5)
      g.map(plt.scatter,"month","modal_price")
      g.add_legend()
```

```
[20]: <seaborn.axisgrid.FacetGrid at 0x7f84d7d776d0>
```


state = Andhra Pradesh
state = Chattisgarh
state = Goa
state = Gujarat
state = Haryana
state = Himachal Pradesh
state = Jammu and Kashmir
state = Jharkhand
state = Karnataka
state = Kerala
state = Madhya Pradesh
state = Maharashtra
state = Nagaland
state = NCT of Delhi
state = Odisha
state = Punjab
state = Rajasthan
state = Telangana
state = Tripura
state = Uttar Pradesh
state = Uttrakhand
state = West Bengal
modal_price
0
5000
10000
15000
20000
month
5
10
variety
Local
Other
Onion
Nasik
Red
White
Beelary-Red
1st Sort
Bangalore-Samall
Puna
Pusa-Red
Bombay (U.P.)
Telagi
Hybrid
Big
Small
2nd Sort
Pole
Dry F.A.Q.
Medium
Bellary

2 Apartat (B): Primeres regressions

[316]:

```
plt.figure(figsize=(30,10))
plt.title("Histograma de l'atribut 0")
plt.xlabel("Attribute Value")
plt.ylabel("Count")

hist = plt.hist(x[:,-1], bins=len( set(x[:,-1])), rwidth=0.2)
plt.gcf().autofmt_xdate()
plt.savefig("images/histModalPrice.png",dpi = 300, bbox_inches = 'tight')
plt.clf()

for i in range(7):
    plt.title("Histograma de l'atribut ")
    plt.xlabel(dataset.columns[i])
    plt.ylabel("Count")
    hist = plt.hist(x[:,i] , bins=len( set(x[:,i])),  rwidth=0.2)
    plt.gcf().autofmt_xdate()
    plt.savefig("images/histogramas/hist" + dataset.columns[i] + ".png", dpi =␣
  ↪300, bbox_inches = 'tight')
    plt.clf()
```

```
<Figure size 2160x720 with 0 Axes>
```

[10]:

```
train_dataset = dataset
train_dataset.drop(['arrival_date'], axis=1, inplace=True)

train_dataset.head()
```

[10]:

```
             state district   market variety  min_price  max_price  \
0  Andhra Pradesh  Kurnool  Kurnool   Local       1350       4390
1  Andhra Pradesh  Kurnool  Kurnool   Local       1390       4400
2  Andhra Pradesh  Kurnool  Kurnool   Local       1460       5150
3  Andhra Pradesh  Kurnool  Kurnool   Local       2010       5200
4  Andhra Pradesh  Kurnool  Kurnool   Local       1320       4050

   modal_price  month  day
0     3100.000      3    1
1     3200.000      4    1
2     4310.000      6    1
3     4200.000      7    1
4     3300.000     10    1
```

[11]:

```
# Categorització de la nostra col·lecció, farem una categorització de state i␣
  ↪variety
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import make_column_transformer
```

```
train_dataset_state_variety = train_dataset[['state','variety', 'min_price',␣
 ↪'max_price', 'month', 'day']]
train_df_x_variety = train_dataset[['variety', 'min_price', 'max_price',␣
 ↪'month', 'day']]
train_df_x_state = train_dataset[['state', 'min_price', 'max_price', 'month',␣
 ↪'day']]
train_df_y = train_dataset['modal_price']

df_x_prices = train_dataset[['min_price', 'max_price']]
df_x_max_price = train_dataset[ 'max_price']


def replace_categorical(df):
    columns = df.columns
    for col in columns:
        if df[col].dtype == 'object':
            df = pd.concat([df, pd.get_dummies(df[col],prefix=col)], axis=1)
            df = df.drop(columns=col)

    return df


train_dataset_state_variety = replace_categorical(train_dataset_state_variety)
train_df_x_variety = replace_categorical(train_df_x_variety)
train_df_x_state = replace_categorical(train_df_x_state)
train_dataset_state_variety
```

[11]:

```
        min_price  max_price  month  day  state_Andhra Pradesh  \
0            1350       4390      3    1                     1
1            1390       4400      4    1                     1
2            1460       5150      6    1                     1
3            2010       5200      7    1                     1
4            1320       4050     10    1                     1
…               …          …      …  …                     …
107100       2200       2300      3    9                     0
107101       2050       2600      4    9                     0
107102       2700       2875      8    9                     0
107103       2625       2875      9    9                     0
107104       2800       2890     10    9                     0

        state_Chattisgarh  state_Goa  state_Gujarat  state_Haryana  \
0                       0          0              0              0
1                       0          0              0              0
2                       0          0              0              0
3                       0          0              0              0
```

```
4                       0          0              0              0
…                   …        …            …            …
107100                  0          0              0              0
107101                  0          0              0              0
107102                  0          0              0              0
107103                  0          0              0              0
107104                  0          0              0              0

        state_Himachal Pradesh  …  variety_Nasik  variety_Onion  \
0                            0  …              0              0
1                            0  …              0              0
2                            0  …              0              0
3                            0  …              0              0
4                            0  …              0              0
…                        …  …            …            …
107100                       0  …              0              0
107101                       0  …              0              0
107102                       0  …              0              0
107103                       0  …              0              0
107104                       0  …              0              0

        variety_Other  variety_Pole  variety_Puna  variety_Pusa-Red  \
0                   0             0             0                 0
1                   0             0             0                 0
2                   0             0             0                 0
3                   0             0             0                 0
4                   0             0             0                 0
…               …           …           …               …
107100              0             0             0                 0
107101              0             0             0                 0
107102              0             0             0                 0
107103              0             0             0                 0
107104              0             0             0                 0

        variety_Red  variety_Small  variety_Telagi  variety_White
0                 0              0               0              0
1                 0              0               0              0
2                 0              0               0              0
3                 0              0               0              0
4                 0              0               0              0
…             …            …             …            …
107100            1              0               0              0
107101            1              0               0              0
107102            1              0               0              0
107103            1              0               0              0
107104            1              0               0              0
```

```
[107105 rows x 47 columns]
```

[12]:

```
import math

def mean_squared_error(y1, y2):
    # comprovem que y1 i y2 tenen la mateixa mida
    assert(len(y1) == len(y2))
    mse = 0
    for i in range(len(y1)):
        mse += (y1[i] - y2[i])**2
    return mse / len(y1)
```

[13]:

```
import numpy as np #importem la llibreria
np.warnings.filterwarnings('ignore')

def mse(v1, v2):
    return ((v1 - v2)**2).mean()
```

[14]:

```
from sklearn.linear_model import LinearRegression

def regression(x, y):
    # Creem un objecte de regressió de sklearn
    regr = LinearRegression()

    # Entrenem el model per a predir y a partir de x
    regr.fit(x, y)

    # Retornem el model entrenat
    return regr
```

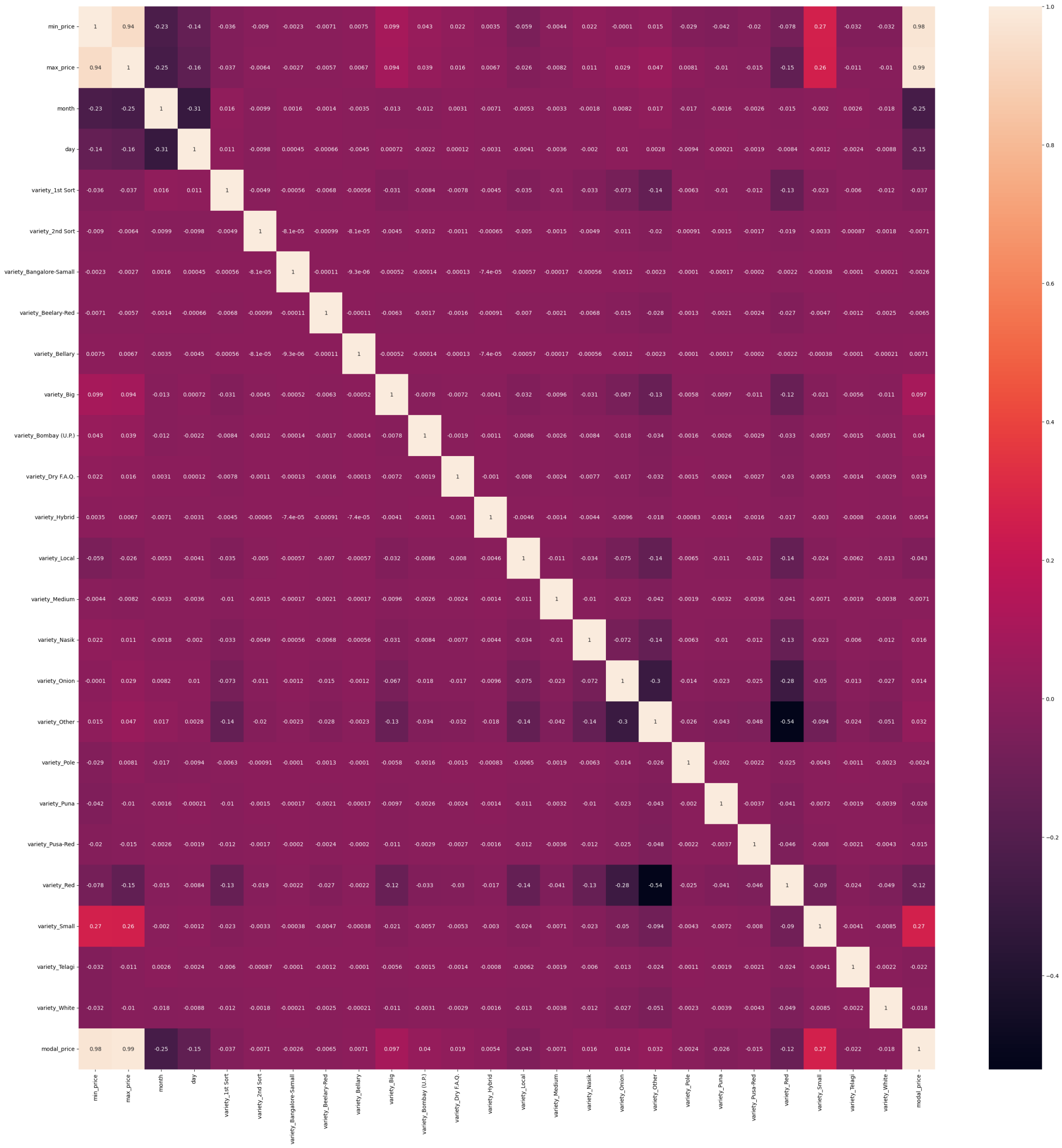
[15]:

```
def standarize(x, mean=None, std=None):
    if mean is None:
        mean = x.mean(0)
    if std is None:
        std = x.std(0)

    return (x - mean[None, :]) / std[None, :], mean, std

train_df_x_norm, mean, std = standarize(train_dataset_state_variety.values)
train_df_x_norm_variety, mean, std = standarize(train_df_x_variety.values)
train_df_x_norm_state, mean, std = standarize(train_df_x_state.values)

train_df_y_norm, mean, std = standarize(train_df_y.values[:, None])
df_x_prices_norm, mean, std = standarize(df_x_prices.values)

#Normalització dels valors
train_df_x_norm
```

```
[15]: array([[-0.37420816,  1.33992296, -0.79899862, …, -0.12535512,
              -0.03278516, -0.06779363],
             [-0.34679915,  1.34631552, -0.46805108, …, -0.12535512,
              -0.03278516, -0.06779363],
             [-0.2988334 ,  1.82575731,  0.19384402, …, -0.12535512,
              -0.03278516, -0.06779363],
             …,
             [ 0.55084567,  0.37145054,  0.85573911, …, -0.12535512,
              -0.03278516, -0.06779363],
             [ 0.49945379,  0.37145054,  1.18668666, …, -0.12535512,
              -0.03278516, -0.06779363],
             [ 0.61936818,  0.38103938,  1.5176342 , …, -0.12535512,
              -0.03278516, -0.06779363]])
```

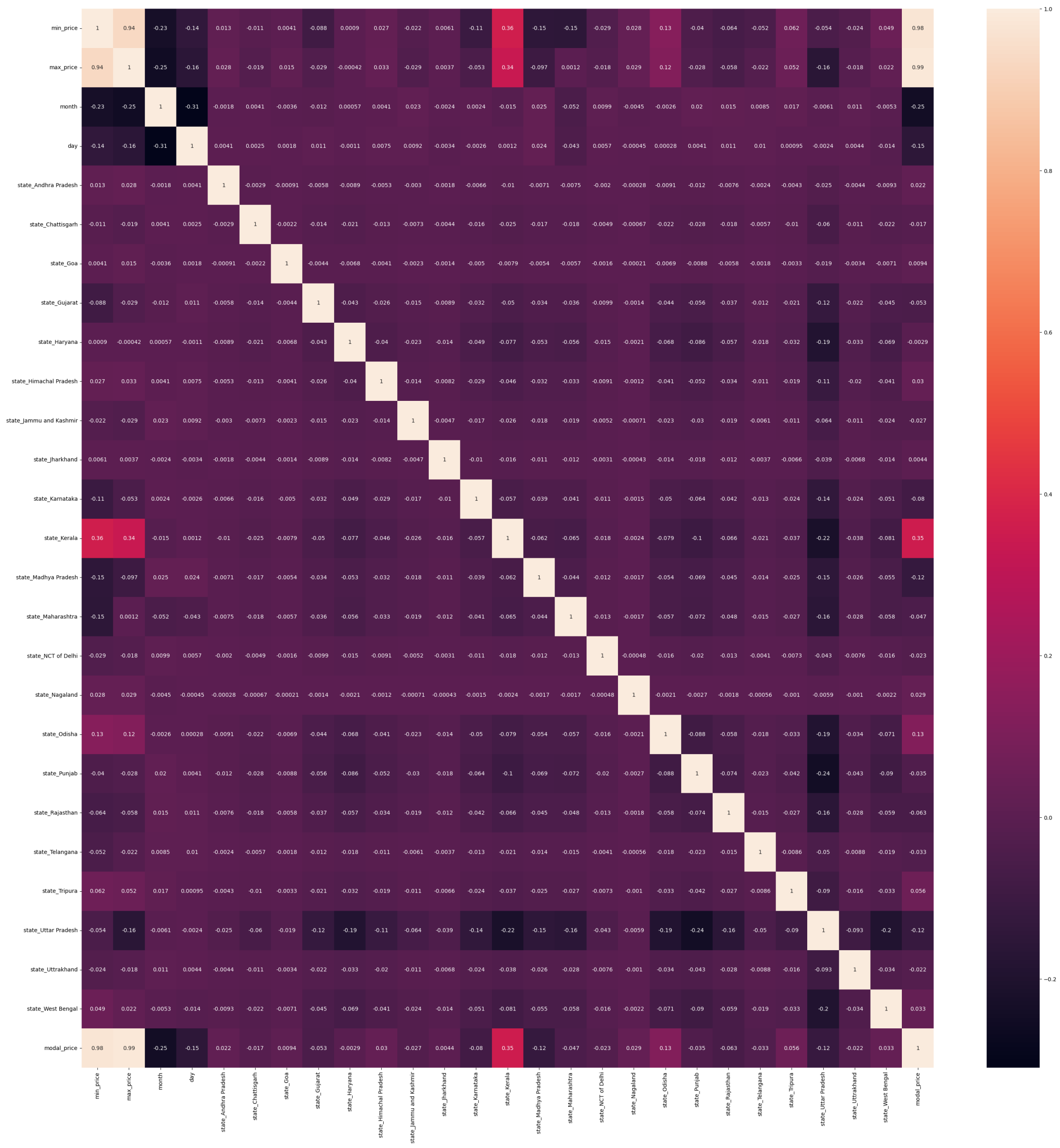
```
[16]: correlation_df_variety_price = pd.concat([train_df_x_variety, train_df_y],␣
        ↪axis=1)
      f, ax = plt.subplots(figsize=(35,35))
      sns.heatmap(correlation_df_variety_price.corr(), annot=True)
```

```
[16]: <AxesSubplot: >
```



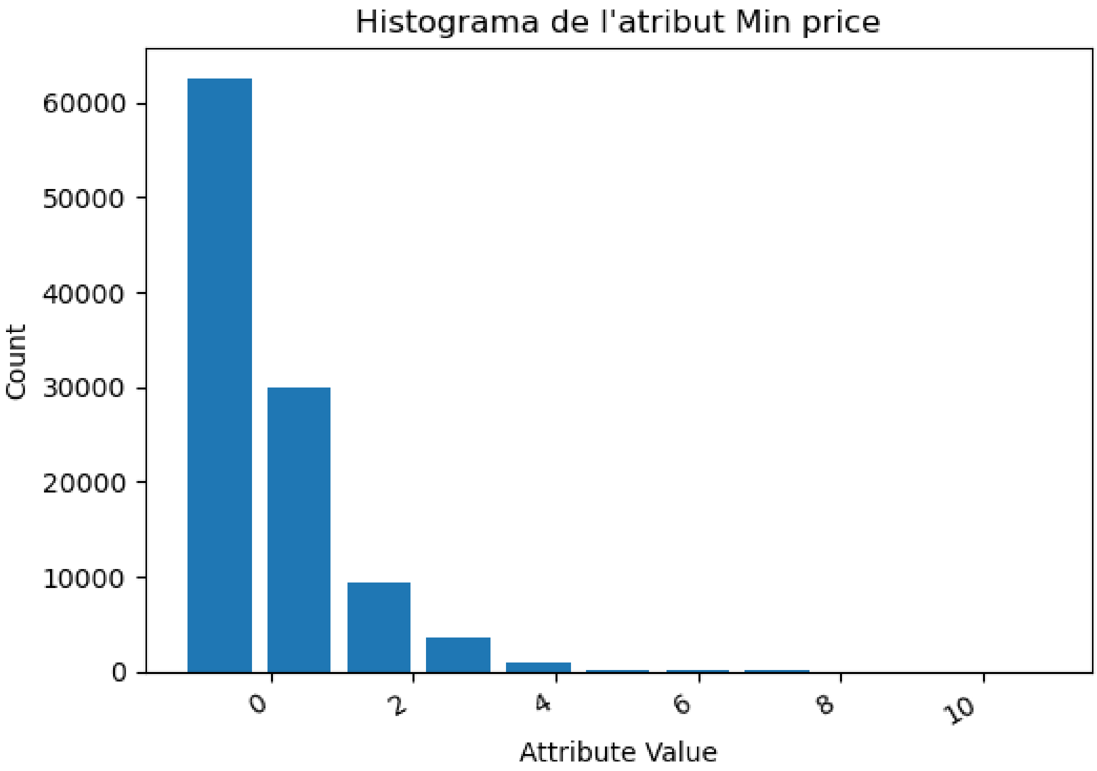
```
[17]: correlation_df_state_price = pd.concat([train_df_x_state, train_df_y], axis=1)
      f, ax = plt.subplots(figsize=(35,35))
      sns.heatmap(correlation_df_state_price.corr(), annot=True)
```

```
[17]: <AxesSubplot: >
```



[18]:

```
plt.figure()
plt.title("Histograma de l'atribut Min price")
plt.xlabel("Attribute Value")
plt.ylabel("Count")

plt.hist(train_df_x_norm[:,0], bins=11, range=[np.min(train_df_x_norm[:,0]), np.
↪max(train_df_x_norm[:,0])], histtype="bar", rwidth=0.8)
plt.gcf().autofmt_xdate()
plt.show()
```



[327]:
```
train_df_y_norm
```

[327]:
```
array([[0.66338973],
       [0.73036101],
       [1.47374221],
       …,
       [0.46247589],
       [0.46247589],
       [0.50935578]])
```

[25]:
```
from sklearn.metrics import r2_score

# Extraiem el primer atribut de x i canviem la mida a #exemples, #dimensions de␣
 ↪l'atribut.
# En el vostre cas, haureu de triar un atribut com a y, i utilitzar la resta␣
 ↪com a x.
mse_array = {}
r2_array = {}
for k in range(47):
    atribut1 = train_df_x_norm[:,k].reshape(train_df_x_norm.shape[0], 1)
    regr = regression(atribut1, train_df_y_norm)
    predicted = regr.predict(atribut1)
    # Mostrem l'error (MSE i R2)
    MSE = mse(train_df_y_norm, predicted)
    mse_array[train_dataset_state_variety.columns[k]] = MSE
    r2 = r2_score(train_df_y_norm, predicted)
    r2_array[train_dataset_state_variety.columns[k]] = r2
```

[37]:

```
#R2 de cada atribut
{j: l for j, l in sorted(r2_array.items(), key=lambda item: item[1])}
```

[37]:

```
{'variety_Pole': 5.545431993714267e-06,
 'variety_Bangalore-Samall': 6.642437629045261e-06,
 'state_Haryana': 8.351419659824444e-06,
 'state_Jharkhand': 1.964720896618921e-05,
 'variety_Hybrid': 2.8982990835868527e-05,
 'variety_Beelary-Red': 4.223113829515679e-05,
 'variety_2nd Sort': 5.0015137394665565e-05,
 'variety_Medium': 5.0306668927468934e-05,
 'variety_Bellary': 5.1022455580529424e-05,
 'state_Goa': 8.880869036964611e-05,
 'variety_Onion': 0.00020322265233696513,
 'variety_Pusa-Red': 0.00022218701167164845,
 'variety_Nasik': 0.00026523454164883997,
 'state_Chattisgarh': 0.0002962059196721656,
 'variety_White': 0.00032334276501055914,
 'variety_Dry F.A.Q.': 0.0003711707712179546,
 'state_Andhra Pradesh': 0.0004634212408722286,
 'state_Uttrakhand': 0.0004769438858474029,
 'variety_Telagi': 0.0004872707182489444,
 'state_NCT of Delhi': 0.0005451567965288895,
 'variety_Puna': 0.000691162845167681,
 'state_Jammu and Kashmir': 0.0007119002360186366,
 'state_Nagaland': 0.0008518466281518533,
 'state_Himachal Pradesh': 0.0008943047030010032,
 'variety_Other': 0.0010535432776486164,
 'state_West Bengal': 0.001076456920363622,
 'state_Telangana': 0.001084883038156459,
 'state_Punjab': 0.0012026663356524692,
 'variety_1st Sort': 0.001385324172781921,
 'variety_Bombay (U.P.)': 0.0015938092662651782,
 'variety_Local': 0.0018742597624321622,
 'state_Maharashtra': 0.0022036653868485745,
 'state_Gujarat': 0.0028497024218138156,
 'state_Tripura': 0.0031357831318789,
 'state_Rajasthan': 0.003961357771358531,
 'state_Karnataka': 0.0064509012902370655,
 'variety_Big': 0.009337523498561762,
 'variety_Red': 0.01360434728652693,
 'state_Uttar Pradesh': 0.014143767491432298,
 'state_Madhya Pradesh': 0.015434614181220083,
 'state_Odisha': 0.015650157829416256,
 'day': 0.023283342752627112,
 'month': 0.061554805146053715,
 'variety_Small': 0.07147300668228285,
 'state_Kerala': 0.12430032554346826,
 'min_price': 0.9508904030415962,
 'max_price': 0.9706636317181121}
```

[38]:
```
#MSE de cada atribut
{k: v for k, v in sorted(mse_array.items(), key=lambda item: item[1])}
```

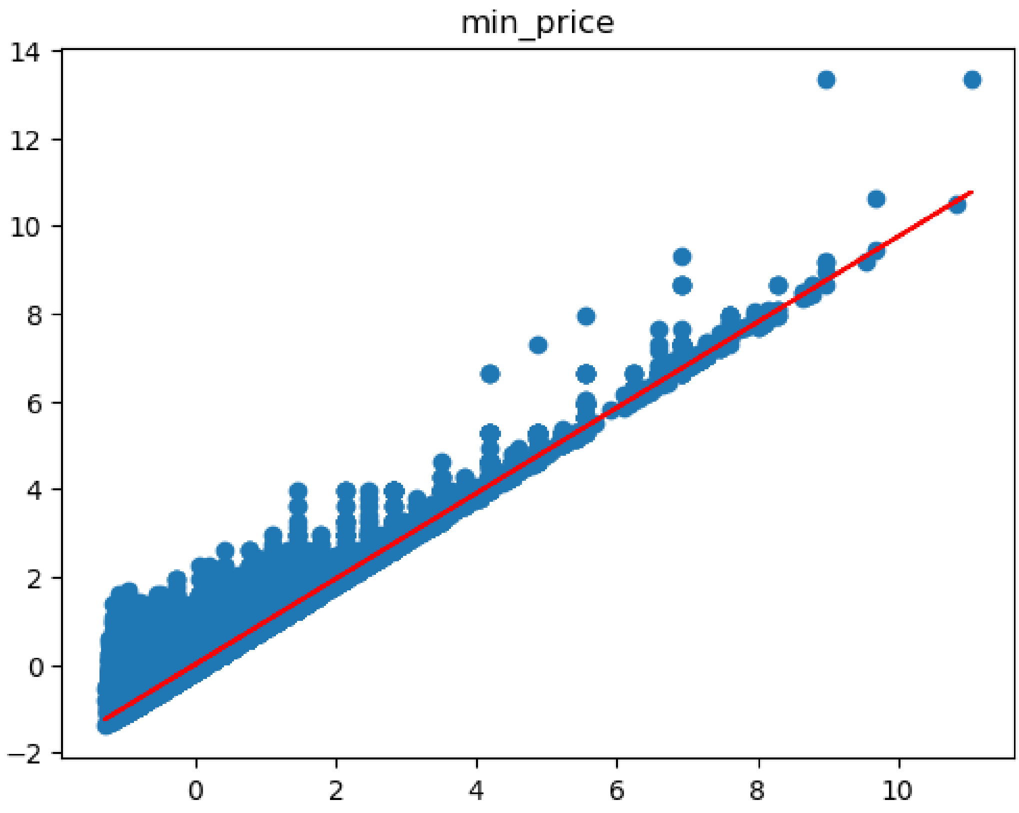
[38]:
```
{'max_price': 0.029336368281887938,
 'min_price': 0.049109596958403744,
 'state_Kerala': 0.8756996744565316,
 'variety_Small': 0.928526993317717,
 'month': 0.9384451948539462,
 'day': 0.9767166572473728,
 'state_Odisha': 0.9843498421705835,
 'state_Madhya Pradesh': 0.9845653858187791,
 'state_Uttar Pradesh': 0.9858562325085668,
 'variety_Red': 0.9863956527134728,
 'variety_Big': 0.9906624765014381,
 'state_Karnataka': 0.9935490987097628,
 'state_Rajasthan': 0.9960386422286412,
 'state_Tripura': 0.996864216868121,
 'state_Gujarat': 0.997150297578186,
 'state_Maharashtra': 0.9977963346131513,
 'variety_Local': 0.9981257402375677,
 'variety_Bombay (U.P.)': 0.9984061907337346,
 'variety_1st Sort': 0.998614675827218,
 'state_Punjab': 0.9987973336643474,
 'state_Telangana': 0.9989151169618434,
 'state_West Bengal': 0.9989235430796363,
 'variety_Other': 0.9989464567223513,
 'state_Himachal Pradesh': 0.9991056952969989,
 'state_Nagaland': 0.999148153371848,
 'state_Jammu and Kashmir': 0.9992880997639813,
 'variety_Puna': 0.9993088371548322,
 'state_NCT of Delhi': 0.999454843203471,
 'variety_Telagi': 0.9995127292817508,
 'state_Uttrakhand': 0.9995230561141525,
 'state_Andhra Pradesh': 0.9995365787591276,
 'variety_Dry F.A.Q.': 0.9996288292287819,
 'variety_White': 0.9996766572349893,
 'state_Chattisgarh': 0.9997037940803277,
 'variety_Nasik': 0.999734765458351,
 'variety_Pusa-Red': 0.9997778129883282,
 'variety_Onion': 0.9997967773476629,
 'state_Goa': 0.9999111913096301,
 'variety_Bellary': 0.9999489775444194,
 'variety_Medium': 0.9999496933310724,
 'variety_2nd Sort': 0.9999499848626052,
 'variety_Beelary-Red': 0.9999577688617047,
 'variety_Hybrid': 0.999971017009164,
 'state_Jharkhand': 0.9999803527910337,
 'state_Haryana': 0.99999164858034,
 'variety_Bangalore-Samall': 0.9999933575623708,
 'variety_Pole': 0.9999944545680062}
```

[34]:

```
#Model per min price
atribut1 = train_df_x_norm[:,0].reshape(train_df_x_norm.shape[0], 1)
regr = regression(atribut1, train_df_y_norm)
predicted = regr.predict(atribut1)

# Mostrem la predicció del model entrenat en color vermell a la Figura anterior 1
plt.figure()
ax = plt.scatter(train_df_x_norm[:,0], train_df_y_norm)
plt.plot(atribut1[:,0], predicted, 'r')
plt.title(train_dataset_state_variety.columns[0])
```

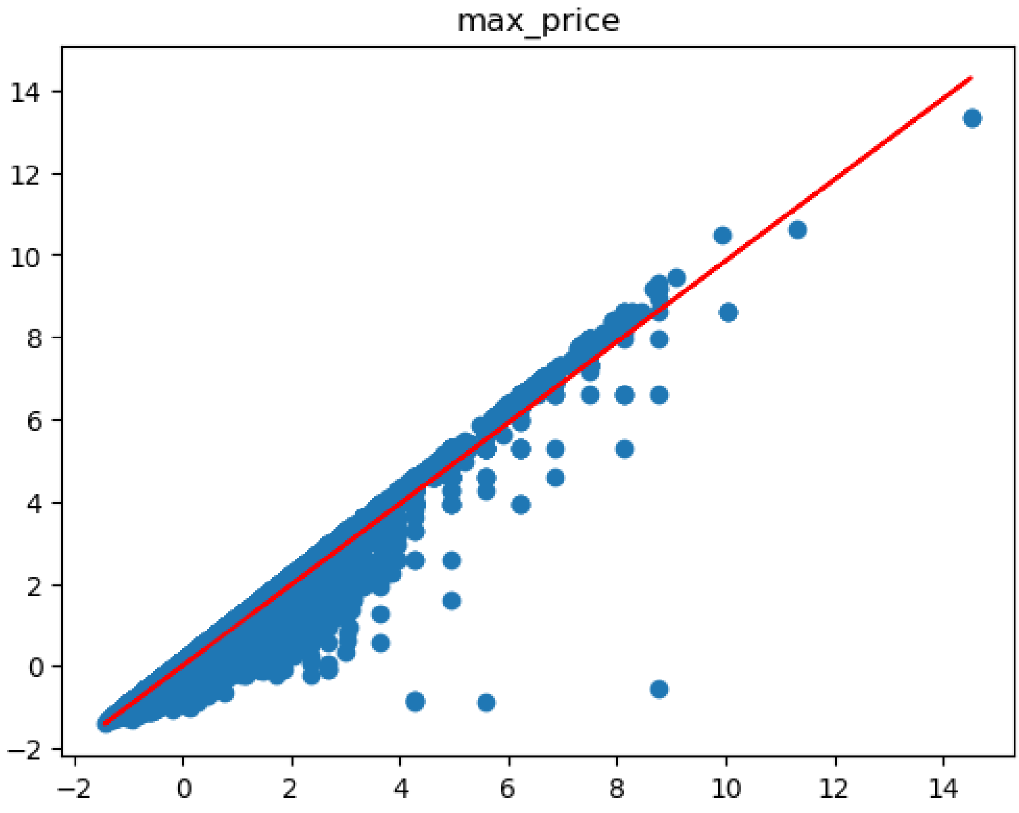
[34]:

```
Text(0.5, 1.0, 'min_price')
```



```
[35]: #Model per el max price
      atribut1 = train_df_x_norm[:,1].reshape(train_df_x_norm.shape[0], 1)
      regr = regression(atribut1, train_df_y_norm)
      predicted = regr.predict(atribut1)

      # Mostrem la predicció del model entrenat en color vermell a la Figura anterior␣
        ↪1
      plt.figure()
      ax = plt.scatter(train_df_x_norm[:,1], train_df_y_norm)
      plt.plot(atribut1[:,0], predicted, 'r')
      plt.title(train_dataset_state_variety.columns[1])
```

```
[35]: Text(0.5, 1.0, 'max_price')
```



[36]:

```
""" Per a assegurar-nos que el model s'ajusta be a dades noves, no vistes,
cal evaluar-lo en un conjunt de validacio (i un altre de test en situacions␣
 ↪reals).
Com que en aquest cas no en tenim, el generarem separant les dades en
un 80% d'entrenament i un 20% de validació.
"""
def split_data(x, y, train_ratio=0.8):
    indices = np.arange(x.shape[0])
    np.random.shuffle(indices)
    n_train = int(np.floor(x.shape[0]*train_ratio))
    indices_train = indices[:n_train]
    indices_val = indices[n_train:]
    x_train = x[indices_train, :]
    y_train = y[indices_train]
    x_val = x[indices_val, :]
    y_val = y[indices_val]
    return x_train, y_train, x_val, y_val

# Dividim dades d'entrenament
x_train, y_train, x_val, y_val = split_data(train_df_x_norm, train_df_y_norm)
```

```
for i in range(x_train.shape[1]):
    x_t = x_train[:,i] # seleccionem atribut i en conjunt de train
    x_v = x_val[:,i] # seleccionem atribut i en conjunt de val.
    x_t = np.reshape(x_t,(x_t.shape[0],1))
    x_v = np.reshape(x_v,(x_v.shape[0],1))

    regr = regression(x_t, y_train)
    error = mse(y_val, regr.predict(x_v)) # calculem error
    r2 = r2_score(y_val, regr.predict(x_v))

    print("Error en atribut %d: %f" %(i, error))
    print("R2 score en atribut %d: %f" %(i, r2))
```

```
Error en atribut 0: 0.046520
R2 score en atribut 0: 0.953231
Error en atribut 1: 0.027465
R2 score en atribut 1: 0.972389
Error en atribut 2: 0.932309
R2 score en atribut 2: 0.062711
Error en atribut 3: 0.968865
R2 score en atribut 3: 0.025960
Error en atribut 4: 0.994218
R2 score en atribut 4: 0.000472
Error en atribut 5: 0.994310
R2 score en atribut 5: 0.000379
Error en atribut 6: 0.994637
R2 score en atribut 6: 0.000050
Error en atribut 7: 0.991773
R2 score en atribut 7: 0.002929
Error en atribut 8: 0.994697
R2 score en atribut 8: -0.000010
Error en atribut 9: 0.993735
R2 score en atribut 9: 0.000958
Error en atribut 10: 0.993893
R2 score en atribut 10: 0.000798
Error en atribut 11: 0.994671
R2 score en atribut 11: 0.000016
Error en atribut 12: 0.988260
R2 score en atribut 12: 0.006461
Error en atribut 13: 0.875188
R2 score en atribut 13: 0.120137
Error en atribut 14: 0.978024
R2 score en atribut 14: 0.016752
Error en atribut 15: 0.992825
R2 score en atribut 15: 0.001872
Error en atribut 16: 0.994021
R2 score en atribut 16: 0.000669
```

```
Error en atribut 17: 0.994109
R2 score en atribut 17: 0.000581
Error en atribut 18: 0.978940
R2 score en atribut 18: 0.015831
Error en atribut 19: 0.994302
R2 score en atribut 19: 0.000387
Error en atribut 20: 0.991211
R2 score en atribut 20: 0.003494
Error en atribut 21: 0.993136
R2 score en atribut 21: 0.001559
Error en atribut 22: 0.991312
R2 score en atribut 22: 0.003393
Error en atribut 23: 0.981122
R2 score en atribut 23: 0.013638
Error en atribut 24: 0.994133
R2 score en atribut 24: 0.000557
Error en atribut 25: 0.993464
R2 score en atribut 25: 0.001230
Error en atribut 26: 0.993527
R2 score en atribut 26: 0.001167
Error en atribut 27: 0.994633
R2 score en atribut 27: 0.000054
Error en atribut 28: 0.994720
R2 score en atribut 28: -0.000033
Error en atribut 29: 0.994702
R2 score en atribut 29: -0.000015
Error en atribut 30: 2477.172201
R2 score en atribut 30: -2489.403717
Error en atribut 31: 0.984265
R2 score en atribut 31: 0.010477
Error en atribut 32: 0.992599
R2 score en atribut 32: 0.002099
Error en atribut 33: 0.994423
R2 score en atribut 33: 0.000266
Error en atribut 34: 0.994650
R2 score en atribut 34: 0.000037
Error en atribut 35: 0.992502
R2 score en atribut 35: 0.002196
Error en atribut 36: 0.994657
R2 score en atribut 36: 0.000030
Error en atribut 37: 0.994625
R2 score en atribut 37: 0.000062
Error en atribut 38: 0.994726
R2 score en atribut 38: -0.000039
Error en atribut 39: 0.993383
R2 score en atribut 39: 0.001311
Error en atribut 40: 0.994705
R2 score en atribut 40: -0.000018
```

```
Error en atribut 41: 0.993812
R2 score en atribut 41: 0.000880
Error en atribut 42: 0.994424
R2 score en atribut 42: 0.000265
Error en atribut 43: 0.982512
R2 score en atribut 43: 0.012240
Error en atribut 44: 0.930124
R2 score en atribut 44: 0.064908
Error en atribut 45: 0.994448
R2 score en atribut 45: 0.000240
Error en atribut 46: 0.994449
R2 score en atribut 46: 0.000240
```

3 Apartat (A): El descens del gradient

[43]:

```
# Cal desnormalitzar les dades
def desnormalitzar(x, mean, std):
    return x * std + mean
```

[44]:

```
import time
from sklearn import linear_model
x_train, y_train, x_val, y_val = split_data(train_df_x_norm, train_df_y_norm)
#x_train, y_train, x_val, y_val = split_data(train_df_x_norm_variety,␣
 ↪train_df_y_norm)
#x_train, y_train, x_val, y_val = split_data(train_df_x_norm_state,␣
 ↪train_df_y_norm)
t1=time.time()
lm = LinearRegression()
lm.fit(x_train,y_train)
print("The intercept term of the linear model:", lm.intercept_)
print("The coefficients of the linear model:", lm.coef_)
t2=time.time()
t_sklearn_linear = float(t2-t1)
print("Time taken: {} seconds".format(t_sklearn_linear))
```

```
The intercept term of the linear model: [-0.00039622]
The coefficients of the linear model: [[ 4.51619911e-01  5.60791899e-01
-1.16353985e-03 -4.73644894e-04
  -1.78585613e+10 -4.27821908e+10 -1.36354044e+10 -8.47735084e+10
  -1.25085067e+11 -7.81363670e+10 -4.54770448e+10 -2.75164604e+10
  -9.56905527e+10 -1.42732091e+11 -1.02219345e+11 -1.07237910e+11
  -3.04884761e+10 -4.19504932e+09 -1.27725277e+11 -1.56119938e+11
  -1.09191170e+11 -3.56185207e+10 -6.34321035e+10 -2.46900139e+11
  -6.54353813e+10 -1.30236934e+11  2.30593010e+11  3.45891406e+10
   3.96903645e+09  4.84147789e+10  3.96903645e+09  2.13945885e+11
   5.93412247e+10  5.48044993e+10  3.14941306e+10  2.36074542e+11
   7.33901292e+10  2.29311620e+11  4.45647180e+11  6.22582136e+11
  4.45263178e+10  7.39213764e+10  8.27074438e+10  6.15246728e+11
  1.60310759e+11  4.25405162e+10  8.76575721e+10]]
Time taken: 0.14576458930969238 seconds
```

[45]:

```
import time
# Regressio lineal per min_price i max_price

x_train, y_train, x_val, y_val = split_data(df_x_prices_norm, train_df_y_norm)
t1=time.time()
lm = LinearRegression()
lm.fit(x_train,y_train)
print("The intercept term of the linear model:", lm.intercept_)
print("The coefficients of the linear model:", lm.coef_)
t2=time.time()
t_sklearn_linear = float(t2-t1)
print("Time taken: {} seconds".format(t_sklearn_linear))
```

```
The intercept term of the linear model: [-0.00016776]
The coefficients of the linear model: [[0.42779028 0.58398615]]
Time taken: 0.008571624755859375 seconds
```

[46]:

```
from sklearn.metrics import mean_squared_error
y_pred = lm.predict(x_val)


y_d = desnormalitzar(y_val, mean, std)
res = desnormalitzar(y_pred, mean, std)
print(mean_squared_error(y_d,res))

recta_x = np.arange(-1,10,0.2)
print(x_val[:,1].size)
print (y_val.size)
print(x_val[:,0].size)

vec = np.vectorize(np.float)

x = np.array([x_val[-1000:,0]])
y = np.array([y_val[-1000:,0]])
z = np.array([x_val[-1000:,1]])

fig = plt.figure()
ax1 = fig.add_subplot(111,projection='3d')
ax1.scatter(x,y,z, c='g',marker='o',alpha=0.6)
y = np.array([y_pred[-1000:,0]])
ax1.scatter(x,y,z, c='b',marker="^",alpha=0.6)
plt.savefig("images/resultats", dpi = 300, bbox_inches = 'tight')
plt.show()
```

```python
plt.clf()
```

```
16445.398709592817
21421
21421
21421
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```