## Корпоративний звіт.Методи аналізу та вибору значущих ознак (Features’ Selection Procedures)

## Легейди Андрія

**Завдання**

Ознайомитись з наданим прикладом використання різних методів відбору значущих ознак (папка Example).

Завантажити файли з даними у папку проекту з посилання:

<https://drive.google.com/file/d/1su22-W8JrRZzm0mea5v8x46YmLh083qp/view?usp=sharing>

Очистити дані та обробити відсутні дані.

Зробити EDA по ознаках.

Проаналізуйте надані дані, використовуючи методи з прикладу та документації, та зберіть результати аналізу у результуючий ранжируваний датафрейм, в якому лівим індексом будуть ознаки, а колонки – результати однофакторного аналізу ознак. Подумайте над системою ранжування такою, яка б врахувала наявність багатьох факторів ранжування (припустимо, що всі вони мають однакову вагу на прийняття вами рішення).

Проаналізуйте ознаки на взаємозалежність, та побудуйте відповідні heatmap засобами seaborn по кожному з використаних методів дослідження.

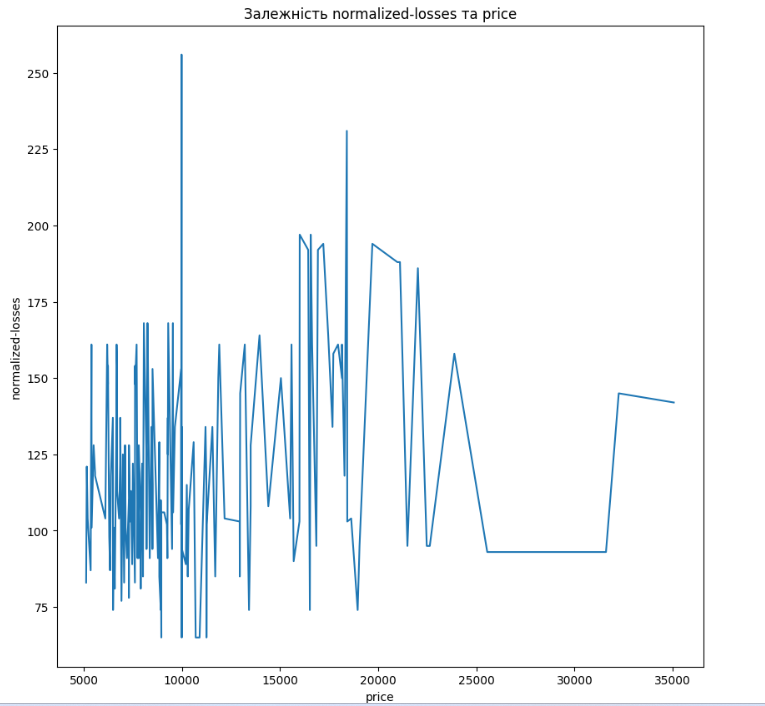
Зберіть висновки у звіт (графіки, висновки текстом у окремому файлі), який потребує належного оформлення, структури тощо.

**Хід роботи**

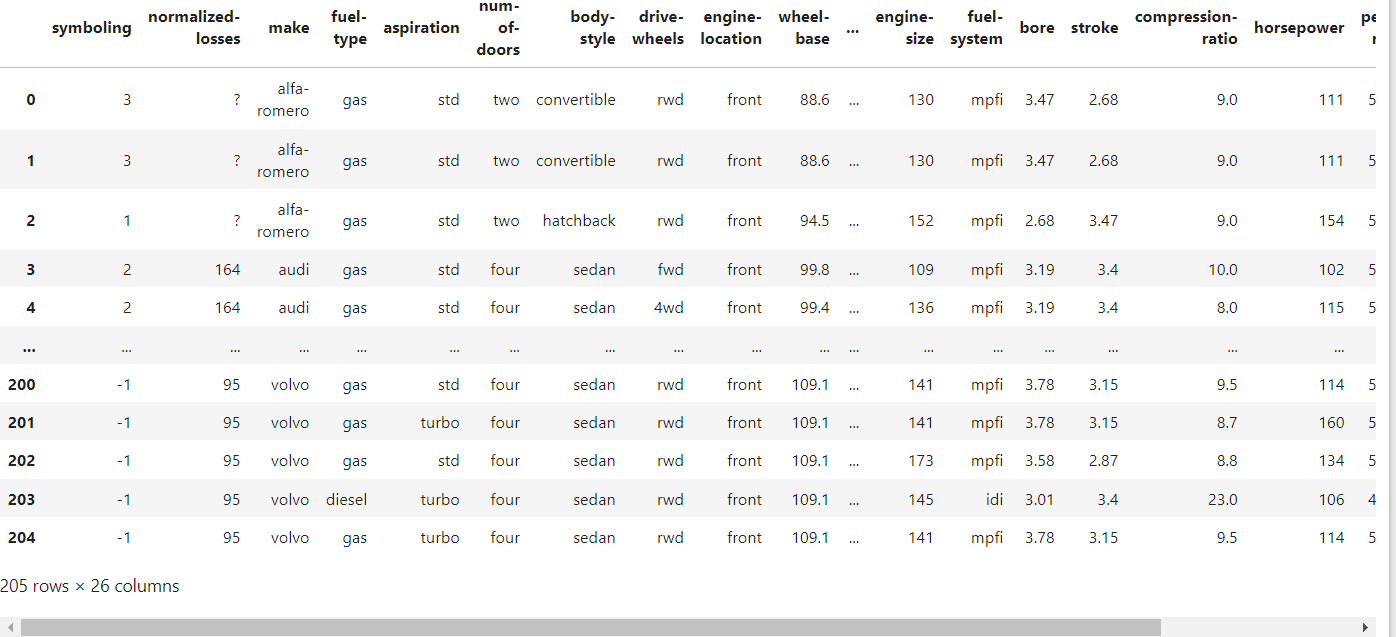
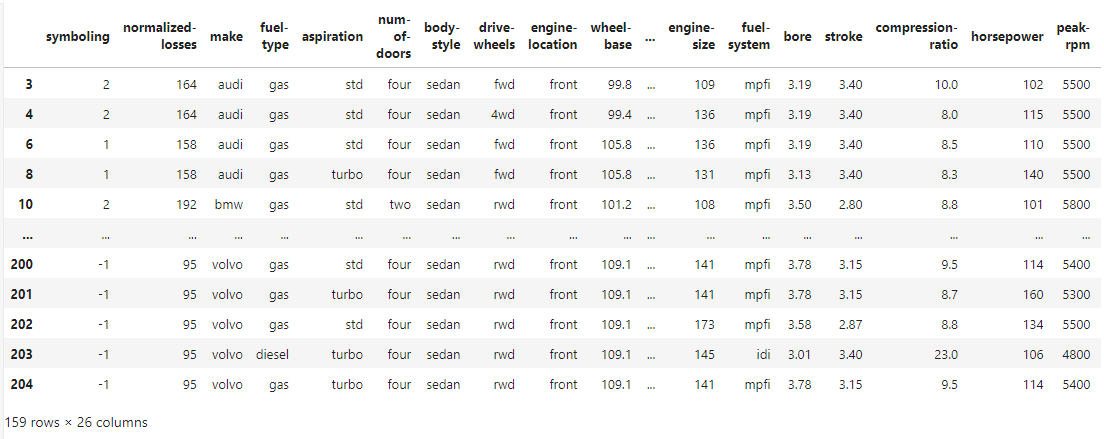
1. Очистили DataFrame від наповних данних. Використовуючи команди заміни та видалення. Бачимо, що кількість стовпців зменшилась.

df2 = df.replace('?', pd.NA, inplace=True)

df.dropna(inplace=True)



1. Залежність normalize-losses до price. Бачимо, що на ціні 10000 найбільше значення normalize-losses



Код:

plt.figure(figsize=[10,10])

df2 = df.sort\_values(by='price')

data1 = df2['price']

data2 = df2['normalized-losses']

plt.plot(data1, data2)

# plot

plt.title('Залежність normalized-losses та price')

plt.xlabel('price')

plt.ylabel('normalized-losses')

plt.show()

Код:

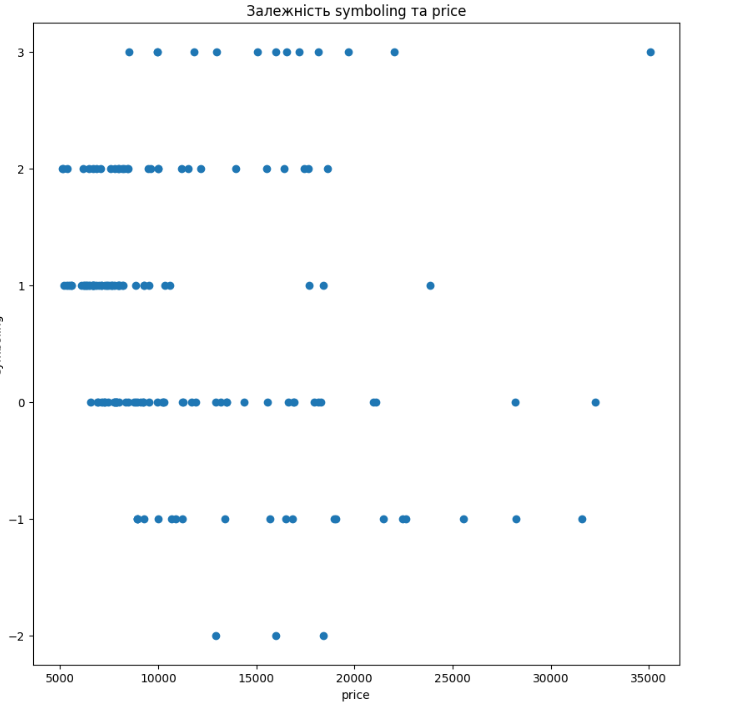
plt.title('Залежність normalized-losses та price')

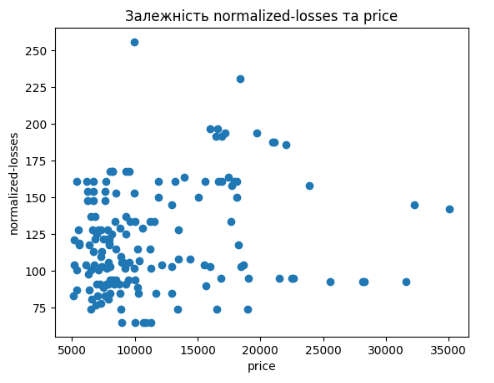
plt.xlabel('price')

plt.ylabel('normalized-losses')

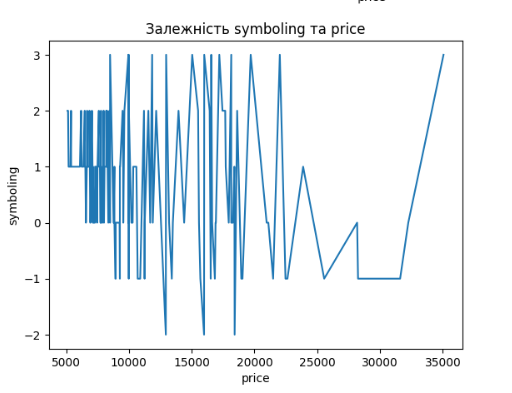
plt.scatter(data1, data2)

plt.show()

1. Залежність symbolik до price. Бачимо як на різних значеннях symbolik змінюється ціна. Значення 3 у symbolik має найбільшу ціну, після нього значення 0, далі -1 і т. д. 



1. Залежність symbolik до price. Бачимо як на різних значеннях symbolik змінюється ціна



Код:

data1 = df2['price']

data2 = df2['symboling']

plt.figure(figsize=[10,10])

plt.title('Залежність symboling та price')

plt.xlabel('price')

plt.ylabel('symboling')

plt.scatter(data1, data2)

plt.show()

plt.plot(data1, data2)

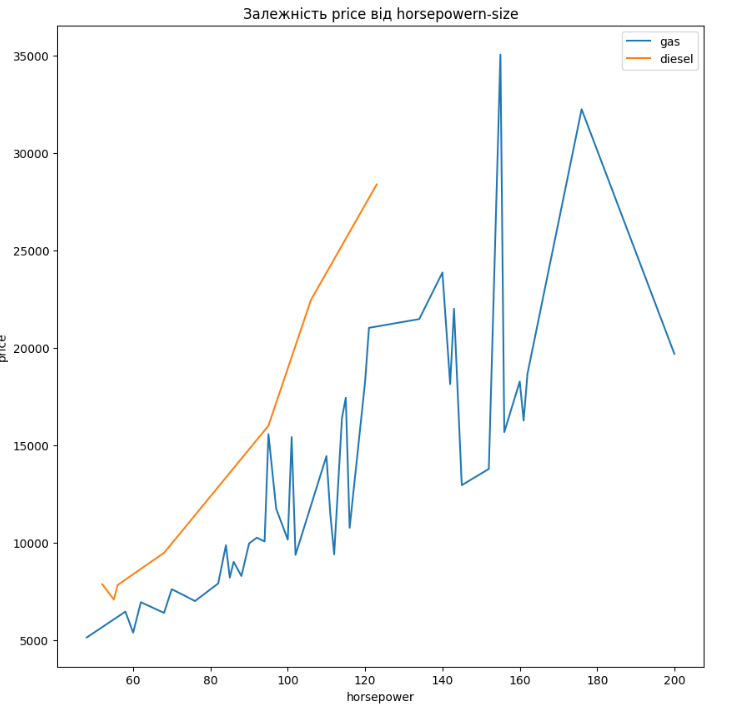
plt.title('Залежність symboling та price')

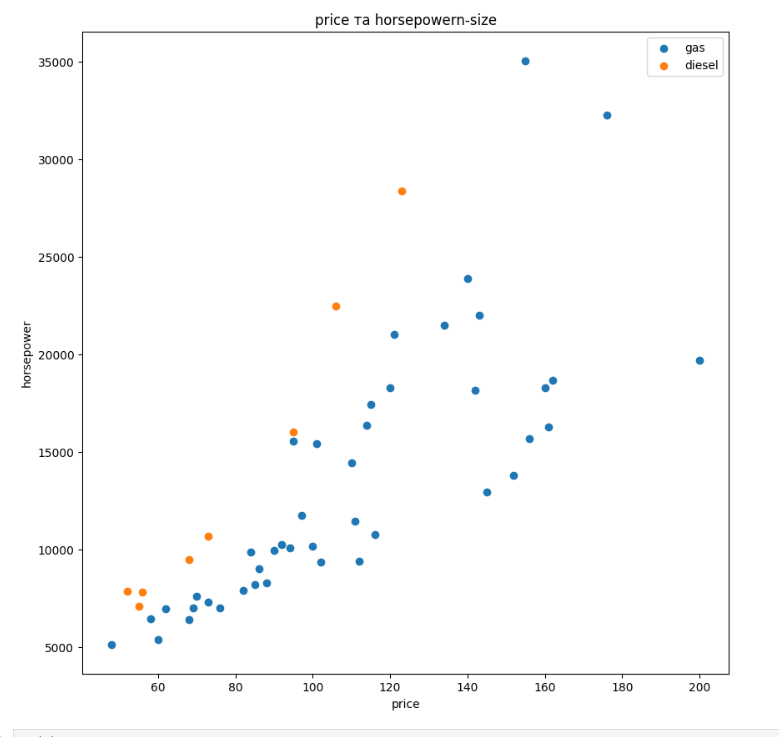
plt.xlabel('price')

plt.ylabel('symboling')

plt.show()

1. Залежність згупованих данних symbolik до price. Бачимо як на різних значеннях symbolik змінюється ціна. Бачимо, що значення gas має меншу кількість в датасеті, та має менший проміжок hoursepower та має менше верхнє значення.





Код

plt.figure(figsize=[10,10])

data\_1=df[ df['fuel-type']=='gas'].groupby(["horsepower"], as\_index=False)["price"].mean()

data\_2=df[ df['fuel-type']=='diesel'].groupby(["horsepower"], as\_index=False)["price"].mean()

plt.figure(figsize=[10,10])

plt.plot(data\_1['horsepower'], data\_1['price'])

plt.plot(data\_2['horsepower'], data\_2['price'])

plt.title('Залежність price від horsepowern-size')

plt.ylabel('price')

plt.xlabel('horsepower')

plt.legend(['gas','diesel'])

plt.show()

plt.figure(figsize=[10,10])

plt.scatter(data\_1['horsepower'], data\_1['price'])

plt.scatter(data\_2['horsepower'], data\_2['price'])

plt.title('price та horsepowern-size')

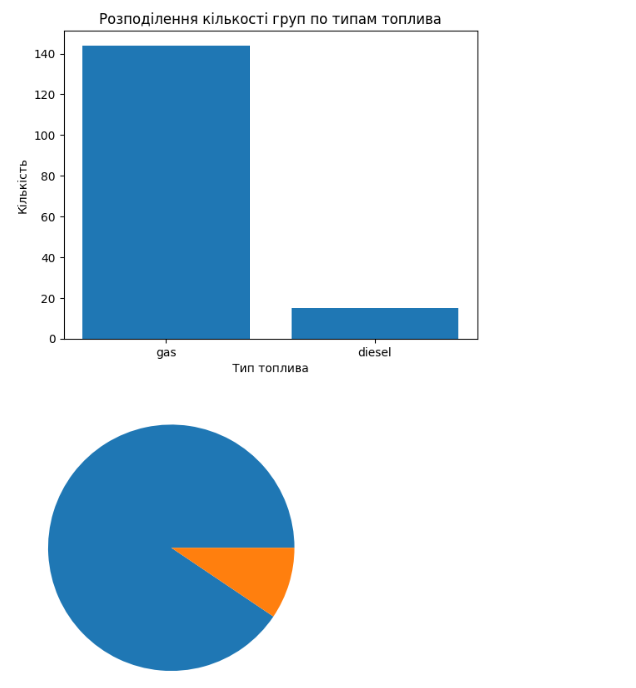
plt.xlabel('price')

plt.ylabel('horsepower')

plt.legend(['gas','diesel'])

plt.show()



1. Бачимо кількість в виборці різних видів топлива на bar та pie діаграмі

Код

# plot

group\_counts = df['fuel-type'].value\_counts()

plt.bar(group\_counts.index, group\_counts)

plt.xlabel('Тип топлива')

plt.ylabel('Кількість')

plt.title('Розподілення кількості груп по типам топлива')

plt.show()

plt.pie(group\_counts)

plt.show()

1. Співвідношення ціни до кількості дверей



Код

data1 =df['num-of-doors']

data2 = df['price']

plt.xlabel('Кількість дверей')

plt.ylabel('Ціна')

plt.title('Відношення кількості дверей до ціни')

plt.scatter(data1, data2)

plt.show()

1. Відображаємо таблицю кореляції. За допомогою

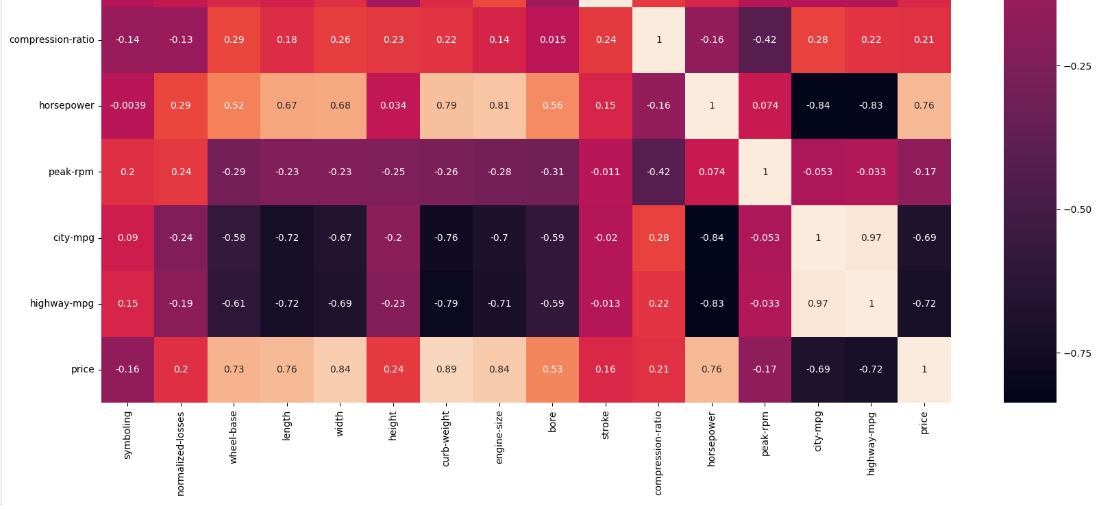
corr\_mat\_df = df.select\_dtypes(include=['float64', 'int64'])

corr\_mat = corr\_mat\_df.corr()

plt.figure(figsize=[20,20])

sns.heatmap(corr\_mat, annot=True)

Та бачимо наскількі різні атрибути відносяться один до одного. Чим більше число тим більше залежність.



Код:

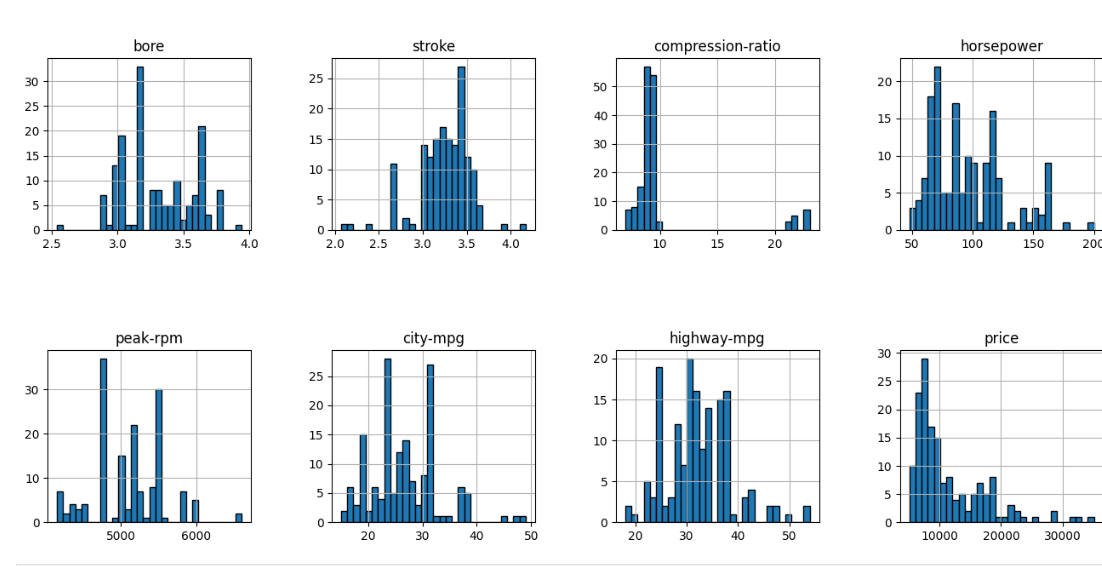
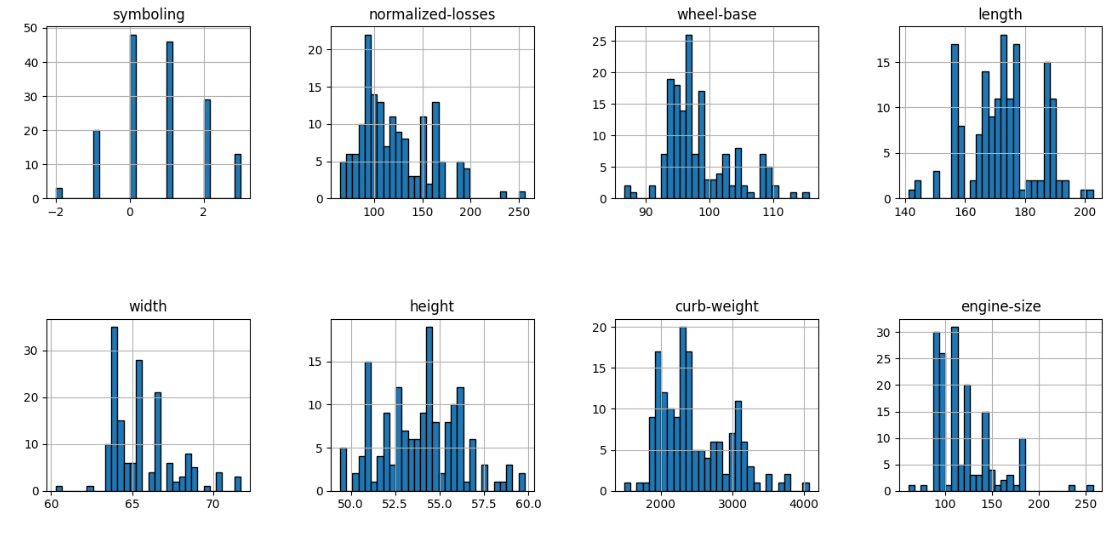
corr\_mat\_df = df.select\_dtypes(include=['float64', 'int64'])

corr\_mat = corr\_mat\_df.corr()

plt.figure(figsize=[20,20])

sns.heatmap(corr\_mat, annot=True)

1. Будуємо subplots\_adjust, який відображає відношення вибраних атрибутів до їх кількості в датасеті.



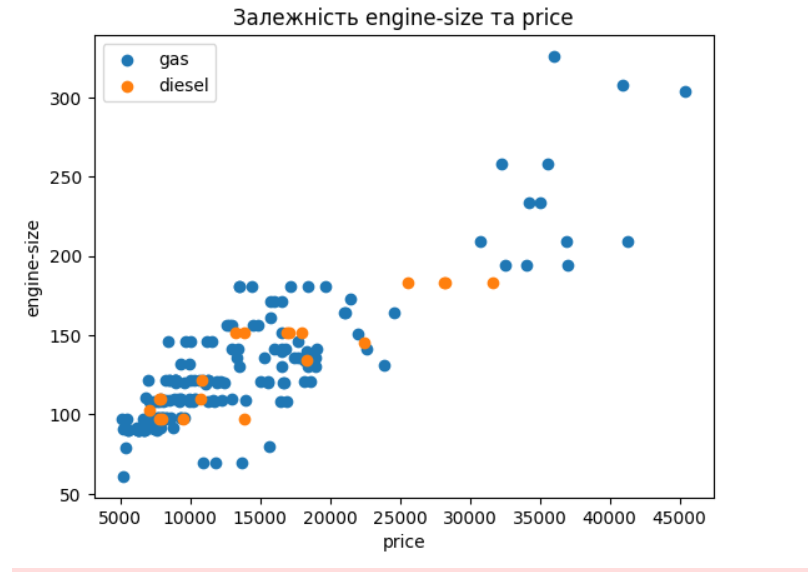
Код:

df.iloc [:, [0,1,9,10,11,12,13,16,18,19,20,21,22,23,24,25]].hist(figsize=(16, 16), bins=30, edgecolor="black")

plt.subplots\_adjust(hspace=0.7, wspace=0.4)

plt.show()

1. Залежність engine-size та price. Бачимо як для різних груп топлива впливає розмір двигуна на ціну



Код:

data10 = df[ df['fuel-type']=='gas']['price']

data11 = df[df['fuel-type']=='diesel']['price']

data20 = df[ df['fuel-type']=='gas']['engine-size']

data21 = df[ df['fuel-type']=='diesel']['engine-size']

plt.title('Залежність engine-size та price')

plt.xlabel('price')

plt.ylabel('engine-size')

plt.scatter(data10, data20, label='gas')

plt.scatter(data11, data21, label='diesel')

plt.legend()

plt.show()

plt.figure(figsize=[20,20])

data=fall[["price","engine-size"]]

sns.jointplot(y = "engine-size", x = "price", kind = 'reg', data = df)

1. На цьому графіку ще бачимо на скільки густо знаходяться значення один від одного
2. Далі виконуємо функції з файлу для виключення найменшої дисперсії з масивів в яких міститься значення кореляції для деяких полів

Код програми:

from sklearn.feature\_selection import VarianceThreshold

# create some synthesized dataset

train\_set = df[["price","engine-size","normalized-losses"]].corr().to\_numpy()

#np.array([[1,2,3],[1,4,7],[1,4,9]]) # the first feature have zero variance

# array([[1, 2, 3],

# [1, 4, 7],

# [1, 4, 9]])

print("train\_set:")

print(train\_set )

test\_set = df[["price","engine-size","symboling"]].corr().to\_numpy()

# np.array([[3,2,3],[1,2,7]]) # the second feature have zero variance

# array([[3, 2, 3],

# [1, 2, 7]])

print("test\_set:")

print(test\_set)

selector = VarianceThreshold()

selector.fit(train\_set) # fit on trainset

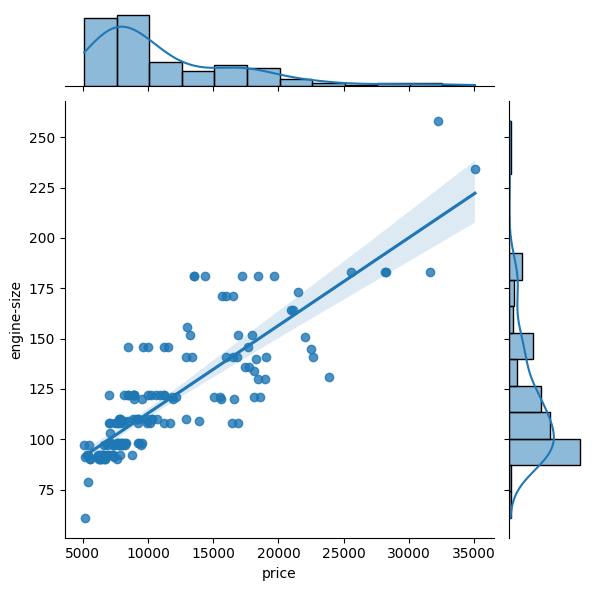
transformed\_train = selector.transform(train\_set) # transform train set

# the first feature has been removed

# array([[2, 3],

# [4, 7],

# [4, 9]])



print("\n\nResult:")

print(transformed\_train)

transformed\_test = selector.transform(test\_set) # transform test set

# array([[2, 3],

# [2, 7]])

# although in the test set the second features has zero variance

# but according to train set, we should remove the first feature only

print(transformed\_test)

Результат виконання:

Так як у масиві немає стовпича з низькою дисперсією, то нічого не видаляється.

train\_set:

[[1. 0.8414956 0.2027613 ]

[0.8414956 1. 0.20781961]

[0.2027613 0.20781961 1. ]]

test\_set:

[[ 1. 0.8414956 -0.16279428]

[ 0.8414956 1. -0.1094533 ]

[-0.16279428 -0.1094533 1. ]]

Result:

[[1. 0.8414956 0.2027613 ]

[0.8414956 1. 0.20781961]

[0.2027613 0.20781961 1. ]]

[[ 1. 0.8414956 -0.16279428]

[ 0.8414956 1. -0.1094533 ]

[-0.16279428 -0.1094533 1. ]]