КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

імені ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра прикладних інформаційних систем

Звіт

до виконання лабораторної роботи №3

з дисципліни « **Data Science та Big Data** »

на тему:

« Python Методи аналізу та вибору значущих ознак (Features' Selection Procedures) »

Виконано:

студ. групи ПП- 41, підгрупа 2

Шкандюк Анною Леонідівною

Перевірено:

к.т.н., доц. Білий Р.О.

Київ – 2023

1. **Мета роботи:**

Метою лабораторної роботи є отримання практичних навичок аналізу та вибору значущих ознак для моделі за допомогою кореляційного аналізу, таблиць сопряження, аналізу багатомірні залежності та дихотомії, дисперсійного аналіз – ANOVA, критерій Хі-квадрат тощо.

2. Контекст:

Ви – data analyst у компанії, яка торгує підтриманими автомобілями по всій Америці (викупає у власника, та перепродає). Ваше керівництво надало вам завдання проаналізувати наявні дані та виявити серед них фактори (ознаки), які впливають на ціну, а також структуру взаємозалежності факторів, та оформити результати дослідження у звіт.

Наданий вам набір даних складається з даних з автомобільного щорічника Ward's Automotive Yearbook за 1985 рік.

Джерела:

· Технічні характеристики імпортних автомобілів і вантажівок моделі 1985 року, автомобільний щорічник Уорда за 1985 рік.

· Personal Auto Manuals, Insurance Services Office, 160 Water Street, New York, NY 10038

· Insurance Collision Report, Insurance Institute for Highway Safety, Watergate 600, Washington, DC 20037.

Цей набір даних складається з трьох типів об’єктів: (а) специфікація автомобіля з точки зору різних характеристик, (б) присвоєний йому рейтинг страхового ризику, (в) його нормалізовані втрати під час використання порівняно з іншими автомобілями. Другий рейтинг відповідає ступеню ризику автомобіля, ніж вказує його ціна. Автомобілям спочатку присвоюється символ фактора ризику, пов'язаний з його ціною. Потім, якщо це більш ризиковано (або менше), цей символ коригується шляхом

переміщення його вгору (або вниз) за шкалою. Актуарії називають цей процес «символізація». Значення +3 вказує на те, що авто є ризикованим, -3, що воно, ймовірно, досить безпечне.

Третім фактором є відносна середня виплата збитку за рік страхування автомобіля. Це значення нормалізовано для всіх автомобілів певної класифікації розміру (дводверні маленькі, універсали, спортивні/спеціальні тощо) і являє собою середні втрати на автомобіль на рік.

Примітка. Кілька атрибутів у базі даних можна використовувати як атрибут «класу».

Інформація про атрибути:

1. symboling: -3, -2, -1, 0, 1, 2, 3
2. normalized-losses: continuous from 65 to 256
3. make: alfa-romero, audi, bmw, chevrolet, dodge, honda, isuzu, jaguar, mazda, mercedes-benz, mercury, mitsubishi, nissan, peugot, plymouth, porsche, renault, saab, subaru, toyota, volkswagen, volvo
4. fuel-type: diesel, gas
5. aspiration: std, turbo
6. num-of-doors: four, two
7. body-style: hardtop, wagon, sedan, hatchback, convertible
8. drive-wheels: 4wd, fwd, rwd
9. engine-location: front, rear
10. wheel-base: continuous from 86.6 120.9
11. length: continuous from 141.1 to 208.1

12. width: continuous from 60.3 to 72.3
13. height: continuous from 47.8 to 59.8
14. curb-weight: continuous from 1488 to 4066
15. engine-type: dohc, dohcv, l, ohc, ohcf, ohcv, rotor
16. num-of-cylinders: eight, five, four, six, three, twelve, two
17. engine-size: continuous from 61 to 326
18. fuel-system: 1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi
19. bore: continuous from 2.54 to 3.94
20. stroke: continuous from 2.07 to 4.17
21. compression-ratio: continuous from 7 to 23
22. horsepower: continuous from 48 to 288
23. peak-rpm: continuous from 4150 to 6600
24. city-mpg: continuous from 13 to 49
25. highway-mpg: continuous from 16 to 54
26. price: continuous from 5118 to 45400.

На базі цього виконати наступні завдання: 1.

3. Хід виконання:

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

2. Завантажити файли з даними у папку проекту з посилання:

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

3. Очистити дані та обробити відсутні дані.

```
import numpy as np

# Explore the correlation between numerical features to identify relationships.
localData = data.copy()
localData.replace('?', np.nan, inplace=True)
localData = localData.apply(pd.to_numeric, errors='coerce')
```

4. Зробити EDA по ознаках.

	symboling	wheel-base	length	width	height
count	205.000000	205.000000	205.000000	205.000000	205.000000
mean	0.834146	98.756585	174.049268	65.907805	53.724878
std	1.245307	6.021776	12.337289	2.145204	2.443522
min	-2.000000	86.600000	141.100000	60.300000	47.800000
25%	0.000000	94.500000	166.300000	64.100000	52.000000
50%	1.000000	97.000000	173.200000	65.500000	54.100000
75%	2.000000	102.400000	183.100000	66.900000	55.500000
max	3.000000	120.900000	208.100000	72.300000	59.800000

	curb-weight	engine-size	compression-ratio	city-mpg	highway-mpg
count	205.000000	205.000000	205.000000	205.000000	205.000000
mean	2555.565854	126.907317	10.142537	25.219512	30.751220
std	520.680204	41.642693	3.972040	6.542142	6.886443
min	1488.000000	61.000000	7.000000	13.000000	16.000000
25%	2145.000000	97.000000	8.600000	19.000000	25.000000
50%	2414.000000	120.000000	9.000000	24.000000	30.000000
75%	2935.000000	141.000000	9.400000	30.000000	34.000000
max	4066.000000	326.000000	23.000000	49.000000	54.000000

symboling: це оцінка ризику для автомобіля в межах страхового портфеля. Від'ємні значення вказують на менший ризик, а позитивні - на більший.

wheel-base: відстань між передніми та задніми колесами автомобіля. Середнє значення - 98.76 дюймів.

length: довжина автомобіля. Середнє значення - 174.05 дюймів.

width: ширина автомобіля. Середнє значення - 65.91 дюймів.

height: висота автомобіля. Середнє значення - 53.72 дюймів.

curb-weight: вага автомобіля без пасажирів та вантажу. Середнє значення - 2555.57 фунтів.

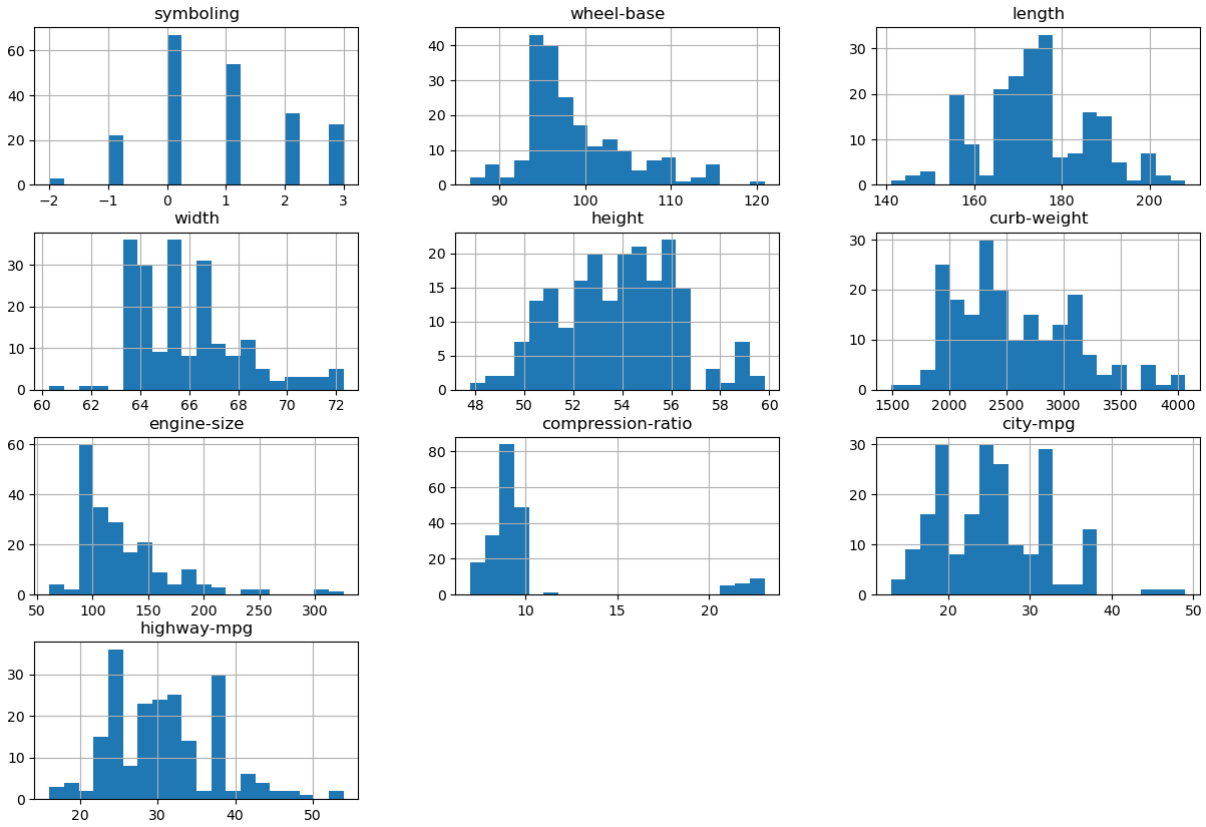
engine-size: об'єм двигуна. Середнє значення - 126.91 кубічних дюймів.

compression-ratio: ступінь стиску двигуна. Середнє значення - 10.14.

ity-mpg: кількість миль, яку автомобіль може проїхати на один галон пального в міських умовах. Середнє значення - 25.22 миль на галон.

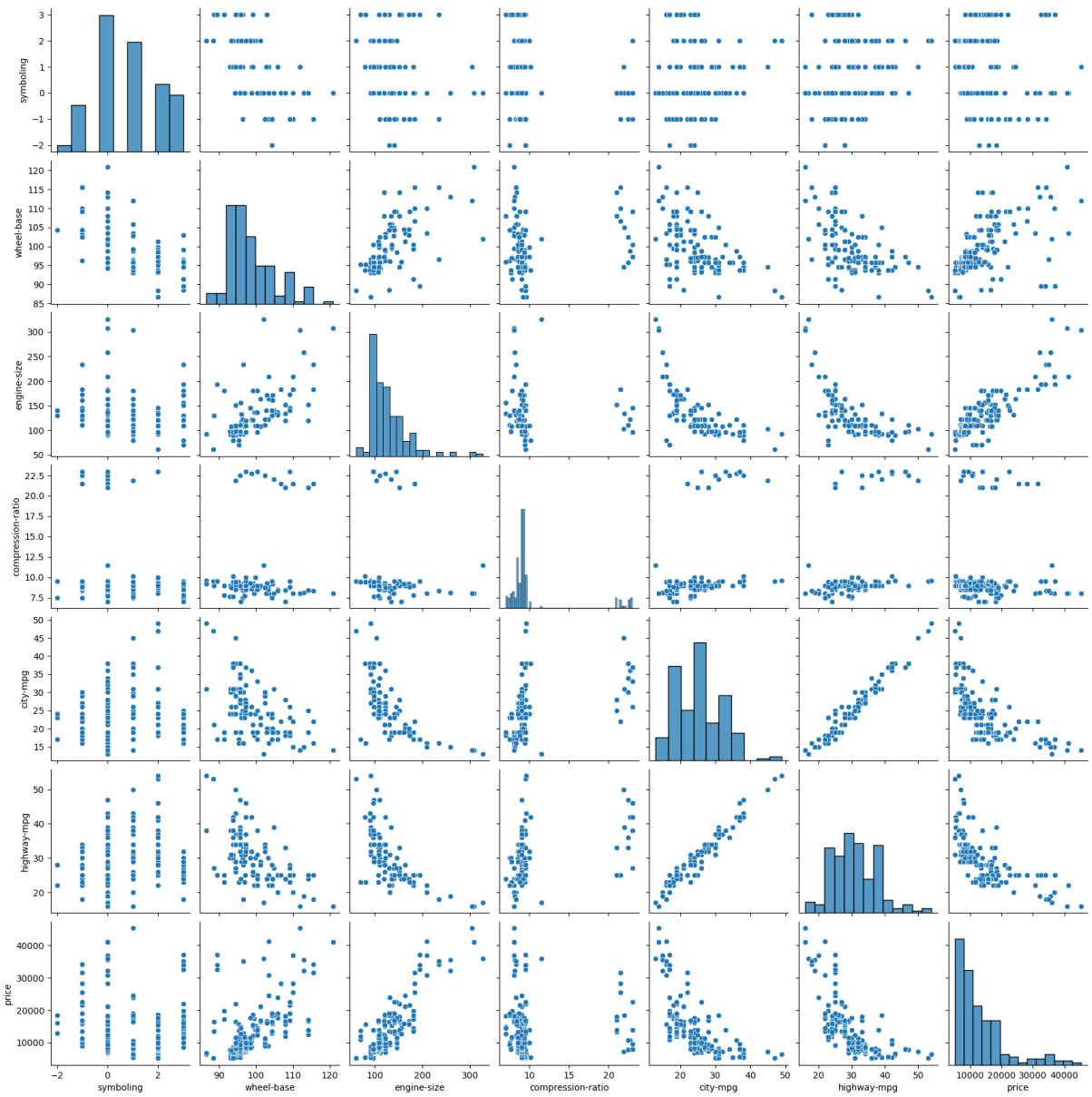
highway-mpg: кількість миль, яку автомобіль може проїхати на один галон пального на шосе. Середнє значення - 30.75 миль на галон.

```
# Visualize the distribution of numerical features to identify patterns and potential outliers.
import matplotlib.pyplot as plt
import seaborn as sns

# Plot histograms for numerical features
data.hist(bins=20, figsize=(15, 10))
plt.show()
```



На цих графіках надані розподіли відповідних числових стовпців.

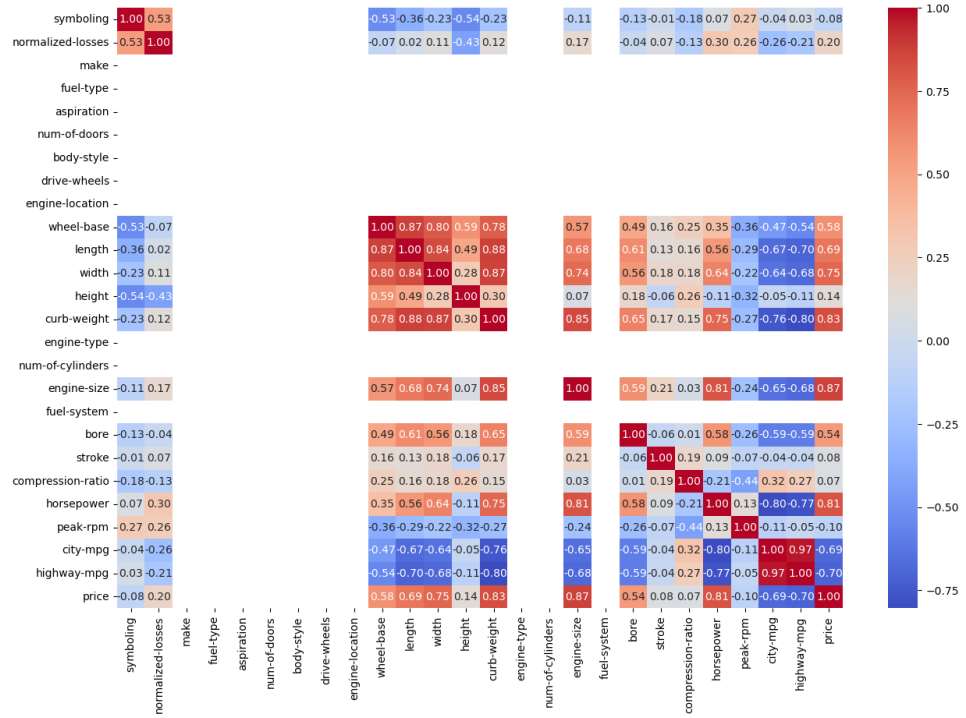
```
# Create a pair plot for selected numerical features to visualize relationships and distributions.
sns.pairplot(localData[numerical_features])
plt.show()
```



5. 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).

Використаємо метод **Distance Correlation** (regression problem) для аналізу наших данних:

```
# Plot heatmap of correlation matrix
plt.figure(figsize=(15, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.show()
```



Це відображається теплова карта матриці кореляції. Звертаючись до цього графіку, можна виявити, як ознаки взаємодіють між собою:

- Ознаки, які позитивно корелюють з ціною: `length` (0.69), `width` (0.75), `curb-weight` (0.83), `engine-size` (0.87), `horsepower` (0.81).

- Ознаки, які негативно корелюють з ціною: `city-mpg` (-0.69), `highway-mpg` (-0.70).

- Є сильна позитивна кореляція між `engine-size` і `horsepower` (0.81), що логічно, оскільки більший об'єм двигуна часто вказує на більшу потужність.

- Є висока кореляція між `length` та `width` (0.75) та `length` та `curb-weight` (0.69), що також є логічним, оскільки більші автомобілі зазвичай будуть довшими та важчими.

- Є сильна негативна кореляція між `city-mpg` та `highway-mpg` (-0.97), що означає, що автомобілі, які ефективні в місті, також ефективні на трасі.

- Є висока кореляція між `width` та `curb-weight` (0.79), що вказує на те, що більші автомобілі можуть бути важчими.

Використаємо метод **f_regression** для відбору кращих ознак:

```
import numpy as np
import pandas as pd
from sklearn.feature_selection import f_regression, SelectKBest

# Замінити "?" на NaN
localData = data.copy()
localData.replace('?', np.nan, inplace=True)

# Перетворити об'єктні колонки на числові (якщо можливо)
localData = localData.apply(pd.to_numeric, errors='ignore')

# Замінити NaN на середнє значення відповідної колонки
for column in localData.columns:
    if localData[column].dtype == 'float64' or localData[column].dtype == 'int64':
        localData[column].fillna(localData[column].mean(), inplace=True)

# Отримати one-hot encoding для категоріальних ознак
localData_encoded = pd.get_dummies(localData, columns=['make', 'fuel-type', 'aspiration', 'r

# Припустимо, що ваш датафрейм називається df
# Замініть 'your_target_column' на назву стовпця, який є вашою цільовою змінною
X = localData_encoded.drop('price', axis=1).values
y = localData_encoded['price'].values

# Розділіть дані на тренувальний та тестовий набори
train_set = X[:150, :]
test_set = X[150:, :]
train_y = y[:150]

# Використовуйте вбудовану функцію f_regression для відбору кращих ознак
selector = SelectKBest(f_regression, k=2)  # k - кількість найкращих ознак для вибору
selector.fit(train_set, train_y)  # підганяємо модель на тренувальному наборі
transformed_train = selector.transform(train_set)  # трансформуємо тренувальний набір
selected_features_indices = selector.get_support(indices=True)  # індекси вибраних ознак

# Виведемо інформацію про вибрані ознаки
print("Selected Features Indices:", selected_features_indices)
print("Selected Features Names:", localData_encoded.columns[selected_features_indices])
print("Scores of Features:", np.nan_to_num(selector.scores_))
```

```
Selected Features Indices: [6 7]
Selected Features Names: Index(['curb-weight', 'engine-size'], dtype='object')
Scores of Features: [4.24962790e-01 2.79899415e+00 7.20271180e+01 1.29454256e+02
 1.57482744e+02 4.12967998e+00 3.08583729e+02 4.73655534e+02
 6.20904809e+01 1.21129454e+00 2.85611662e+00 2.06752881e+02
 6.36065358e+00 1.15925657e+02 1.37508196e+02 1.13716298e-01
 1.10344993e+00 1.87876517e+01 2.46738588e+00 4.52486681e+00
 6.10662864e+00 4.05231315e-01 1.94984929e+01 2.53082113e+00
 6.12082187e+01 9.51310324e-02 3.97496852e+00 3.13347803e+00
 4.37095219e-01 3.34186431e+00 1.43842251e+01 4.70397368e-01
 1.62753544e-01 4.86970845e+00 0.00000000e+00 0.00000000e+00
 0.00000000e+00 4.84674890e+00 4.84674890e+00 2.19551805e+00
 2.19551805e+00 3.56336038e-01 1.96149287e-01 7.72016398e+00
 1.91481832e+01 1.67496484e+01 3.61175332e+00 5.85544787e-01
 6.49898378e-01 1.02025977e+02 1.31663090e+02 1.93460495e+01
 1.93460495e+01 6.30087259e+00 4.76344570e-03 1.13632618e-01
 1.66541317e+01 1.10439412e-03 2.69695260e+01 3.33984353e-02
 3.27402612e+01 9.98654525e+00 1.46006035e+02 4.89858217e+01
 9.89402148e-01 6.74889983e+00 3.33984353e-02 6.30375133e+00
 5.61550773e+01 1.10575800e-01 4.84674890e+00 9.38109198e-03
 7.26428520e+01 9.97744934e-01 1.00062429e-01]
```

Виведено індекси та назви вибраних ознак разом із їхніми балами (scores):

Selected Features Indices: це індекси вибраних ознак у вихідному наборі даних.

Selected Features Names: це назви вибраних ознак. У нашому випадку, це 'curb-weight' та 'engine-size'.

Scores of Features: це оцінки кожної ознаки за їхньою важливістю для моделі. Чим вище бал, тим важливіше.

З урахуванням результатів:

- "curb-weight" та "engine-size" є вибраними ознаками, які модель вважає найважливішими для прогнозування цільової змінної.

- Інші ознаки мають свої власні бали, і їх можна також використовувати для визначення їх важливості в контексті моделі.

Використаємо метод Mutual Information (regression problem)

```python
import numpy as np
import pandas as pd
from sklearn.feature_selection import mutual_info_regression, SelectKBest

# Замінити "?" на NaN
localData = data.copy()
localData.replace('?', np.nan, inplace=True)

# Перетворити об'єктні колонки на числові (якщо можливо)
localData = localData.apply(pd.to_numeric, errors='ignore')

# Замінити NaN на середнє значення відповідної колонки
for column in localData.columns:
    if localData[column].dtype == 'float64' or localData[column].dtype == 'int64':
        localData[column].fillna(localData[column].mean(), inplace=True)

# Отримати one-hot encoding для категоріальних ознак
localData_encoded = pd.get_dummies(localData, columns=['make', 'fuel-type', 'aspiration', 'num-of-doors'

# Замініть 'price' на вашу цільову змінну
X = localData_encoded.drop('price', axis=1).values
y = localData_encoded['price'].values

# Створіть функцію для взаємної інформації (MI)
def udf_MI(X, y):
    result = mutual_info_regression(X, y, n_neighbors=5)  # встановіть кількість сусідів за потребою
    return result

# Використовуйте вбудовану функцію SelectKBest для відбору кращих ознак
selector = SelectKBest(udf_MI, k=2)  # k - кількість найкращих ознак для вибору
selector.fit(X, y)  # підганяємо модель
selected_features_indices = selector.get_support(indices=True)  # індекси вибраних ознак

# Виведемо інформацію про вибрані ознаки
print("Selected Features Indices:", selected_features_indices)
print("Selected Features Names:", localData_encoded.columns[selected_features_indices])
print("Scores of Features:", selector.scores_)
```

```
Selected Features Indices: [ 6 14]
Selected Features Names: Index(['curb-weight', 'highway-mpg'], dtype='object')
Scores of Features: [1.55014839e-01 1.65123303e-01 4.54736030e-01 5.51438423e-01
 6.26235817e-01 2.17691641e-01 8.85573043e-01 7.43795119e-01
 4.00986471e-01 2.87822195e-01 6.47650297e-02 7.56937042e-01
 1.38550190e-01 6.95132259e-01 7.94287456e-01 1.71676408e-03
 3.10910441e-02 3.46439326e-02 9.03755951e-04 1.43195636e-02
 3.84436002e-02 0.00000000e+00 0.00000000e+00 1.16404527e-02
 9.35789182e-02 0.00000000e+00 9.82190396e-03 2.15699019e-02
 4.98099713e-02 1.11182803e-02 0.00000000e+00 9.51697752e-04
 1.50926040e-02 3.10665117e-02 1.32427168e-02 1.68420510e-02
 5.55783997e-02 1.02305570e-02 1.02305570e-02 7.28654901e-02
 7.28654901e-02 1.38290149e-02 1.09966731e-02 4.56305781e-03
 0.00000000e+00 8.63636742e-02 4.45545104e-02 2.43571599e-02
 7.00076186e-03 2.76001286e-01 2.92139154e-01 0.00000000e+00
 0.00000000e+00 3.28632283e-02 8.88178420e-16 4.37053691e-02
 9.23622871e-02 4.44057979e-03 4.23240068e-02 0.00000000e+00
 1.28699003e-03 5.57134552e-02 2.77382850e-01 1.30326571e-01
 1.95121951e-03 0.00000000e+00 0.00000000e+00 4.50325622e-02
 3.02037428e-01 0.00000000e+00 1.02305570e-02 9.75609756e-04
 2.91843461e-01 9.42390674e-03 0.00000000e+00]
```

Ці дві ознаки (curb-weight та highway-mpg) мають найвищі оцінки взаємної інформації та були вибрані як кращі для використання у вашій моделі.

Використаємо метод **Chi-squared Statistics** (classification problem)

```python
import numpy as np
import pandas as pd
from sklearn.feature_selection import chi2, SelectKBest

# Замінити "?" на NaN
localData = data.copy()
localData.replace('?', np.nan, inplace=True)

# Перетворити об'єктні колонки на числові (якщо можливо)
localData = localData.apply(pd.to_numeric, errors='ignore')

# Замінити NaN на середнє значення відповідної колонки
for column in localData.columns:
    if localData[column].dtype == 'float64' or localData[column].dtype == 'int64':
        localData[column].fillna(localData[column].mean(), inplace=True)

# Отримати one-hot encoding для категоріальних ознак
localData_encoded = pd.get_dummies(localData, columns=['make', 'fuel-type', 'aspiration', 'num-of-doo

# Замініть 'price' на назву стовпця, який є вашою цільовою змінною
X = localData_encoded.drop('symboling', axis=1).values
y = localData_encoded['symboling'].values

# Розділіть дані на тренувальний та тестовий набори
train_set = X[:150, :]
test_set = X[150:, :]
train_y = y[:150]

from sklearn.preprocessing import MinMaxScaler

# Масштабуємо дані в інтервал від 0 до 1
scaler = MinMaxScaler()
train_set_scaled = scaler.fit_transform(train_set)

# Використовуйте вбудовану функцію chi2 для відбору кращих ознак
selector = SelectKBest(chi2, k=2)
selector.fit(train_set_scaled, train_y)
transformed_train = selector.transform(train_set_scaled)
selected_features_indices = selector.get_support(indices=True)

# Виведемо інформацію про вибрані ознаки
print("Selected Features Indices:", selected_features_indices)
print("Selected Features Names:", localData_encoded.columns[selected_features_indices])
print("Scores of Features:", selector.scores_)
```

```
Selected Features Indices: [24 42]
Selected Features Names: Index(['make_mazda', 'num-of-doors_four'], dtype='object')
Scores of Features: [ 1.62658939  5.45219979  3.1706397   1.68357442  5.42992238  4.6570107
  2.47973733  2.89062523  0.28030511  4.70609481  3.23952126  2.32299458
  3.89031789  3.24852857  5.13715188  6.75948196  7.6505448   5.88577828
  1.62133466 11.48147659  3.82722217  1.54933433  5.33333333  4.21945911
 37.46018531  2.19148936 10.58373162  5.476393   19.55555556  8.3910778
 16.50786309  3.33625731 15.62471396 12.67105263         nan         nan
         nan 19.15220083  1.66540877  1.74631356  7.28634278 30.4970631
 34.07148277 22.08695652  2.9901921  25.9849284  17.08585267 15.80539166
  7.43191312 10.85138499 17.7000331   0.33806566 16.56521739  6.95025002
  2.19148936 16.667154    8.97034735  8.64988558  2.85767269 22.08695652
  3.69880916 16.42706056  2.64669776  8.08581446  6.89473684  1.77777778
 22.08695652  3.57294106 17.77332416 16.56521739 19.15220083  5.52173913
  7.33393092 14.68838377  6.89473684]
```

Ці дві ознаки (make_mazda та num-of-doors_four) мають найвищі оцінки взаємної інформації та були вибрані як кращі для використання у вашій моделі.

Зберємо результати аналізу у результуючий ранжируваний датафрейм, в якому лівим індексом будуть ознаки, а колонки – результати однофакторного аналізу ознак.

```
import numpy as np
import pandas as pd
from sklearn.feature_selection import chi2, f_regression, mutual_info_regression, SelectKBest
from minepy import MINE
from sklearn.preprocessing import MinMaxScaler

# Замінити "?" на NaN
localData = data.copy()
localData.replace('?', np.nan, inplace=True)

# Перетворити об'єктні колонки на числові (якщо можливо)
localData = localData.apply(pd.to_numeric, errors='ignore')

# Замінити NaN на середнє значення відповідної колонки
for column in localData.columns:
    if localData[column].dtype == 'float64' or localData[column].dtype == 'int64':
        localData[column].fillna(localData[column].mean(), inplace=True)

# Отримати one-hot encoding для категоріальних ознак
localData_encoded = pd.get_dummies(localData, columns=['make', 'fuel-type', 'aspiration', 'num-of-doors', 'body-style', 'drive-w

# Замініть 'symboling' на назву стовпця, який є вашою цільовою змінною
X = localData_encoded.drop('symboling', axis=1).values
y = localData_encoded['symboling'].values

# Розділіть дані на тренувальний та тестовий набори
train_set = X[:150, :]
test_set = X[150:, :]
train_y = y[:150]

# Масштабуємо дані в інтервал від 0 до 1
scaler = MinMaxScaler()
train_set_scaled = scaler.fit_transform(train_set)

# Визначаємо метод MINE для distance correlation
def mic(x, y):
    m = MINE()
    m.compute_score(x, y)
    return m.mic()
```

```
# Використовуйте метод Distance Correlation
distance_corr_scores = [mic(train_set_scaled[:, i], train_y) for i in range(train_set_scaled.shape[1])]

# Використовуйте метод f_regression
f_regression_scores, _ = f_regression(train_set_scaled, train_y)

# Використовуйте метод Mutual Information
mutual_info_scores = [mutual_info_regression(train_set_scaled[:, i].reshape(-1, 1), train_y) for i in range(train_set_scaled.shap

# Використовуйте метод Chi-squared Statistics
chi2_scores, _ = chi2(train_set_scaled, train_y)

# Створимо результуючий ранжированый датафрейм
result_df = pd.DataFrame({
    'Distance Correlation': distance_corr_scores,
    'f_regression': f_regression_scores,
    'Mutual Information': mutual_info_scores,
    'Chi-squared Statistics': chi2_scores
}, index=localData_encoded.drop('symboling', axis=1).columns)  # Виключимо цільову змінну

# Використайте iloc для визначення індексів колонок для сортування
sorted_columns = result_df.mean(axis=1).sort_values(ascending=False).index
result_df = result_df.loc[sorted_columns]

# Виведемо ранжированый датафрейм
print(result_df)
```

Результат:

```
                  Distance Correlation  f_regression  \
num-of-doors_two                  0.384014    103.055172
num-of-doors_four                 0.371306     96.744487
```

height	0.522722	72.649362
body-style_hatchback	0.224095	49.687255
wheel-base	0.626518	57.830820
...	...	...
make_isuzu	0.010890	0.166949
make_chevrolet	0.009321	0.000939
make_toyota	0.000000	NaN
make_volvo	0.000000	NaN
make_volkswagen	0.000000	NaN

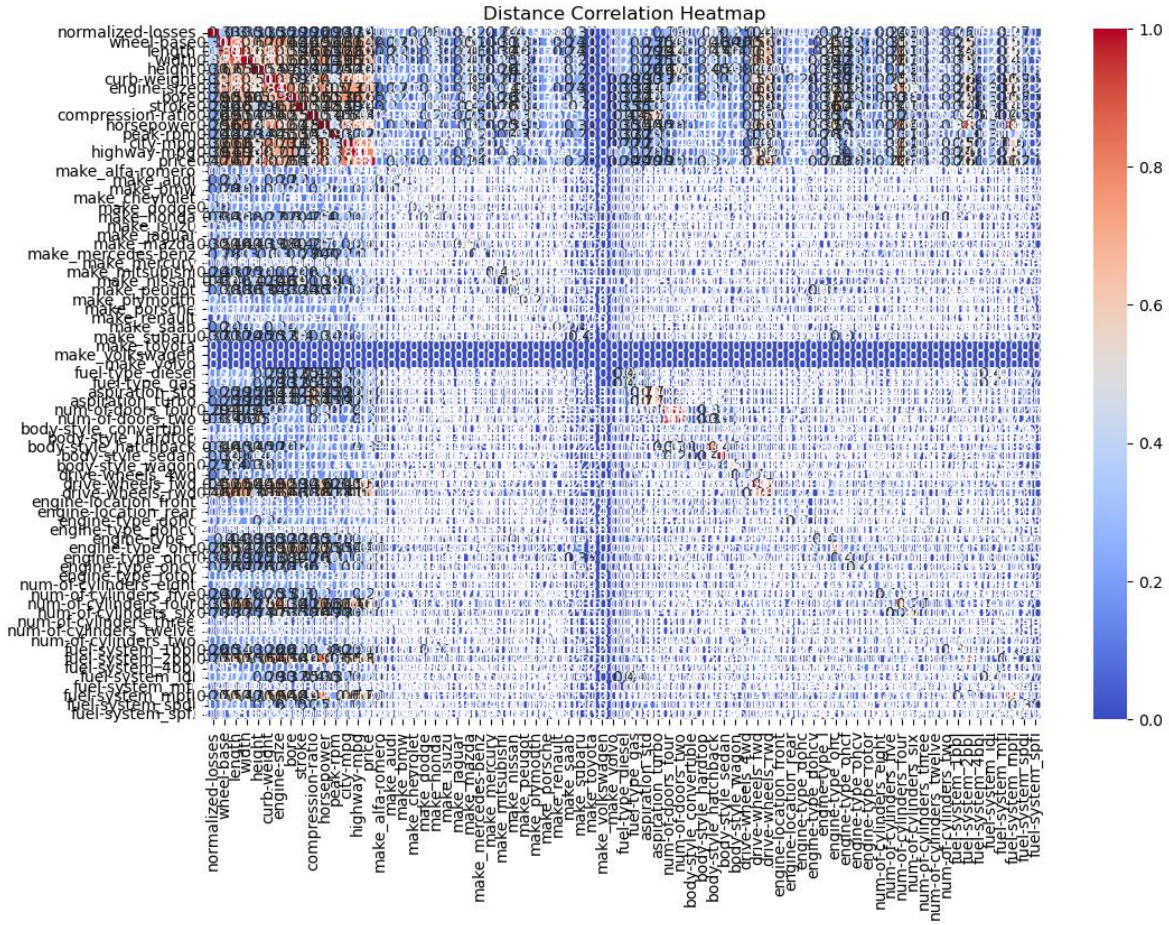
	Mutual Information	Chi-squared Statistics
num-of-doors_two	[0.2985313887321164]	34.071483
num-of-doors_four	[0.2503499032315002]	30.497063
height	[0.5625562916977223]	5.429922
body-style_hatchback	[0.13365811625345225]	25.984928
wheel-base	[0.633686890852061]	5.452200
...	...	...
make_isuzu	[0]	1.549334
make_chevrolet	[0.06444140088981776]	1.621335
make_toyota	[0.006111628734789765]	NaN
make_volvo	[0]	NaN
make_volkswagen	[0]	NaN

Отже, як бачимо, результати аналізу збираються в результуючий ранжируваний датафрейм, в якому лівим індексом є ознаки, а колонки відображають різні результати однофакторного аналізу, такі як Distance Correlation, f_regression, Mutual Information, і Chi-squared Statistics.

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

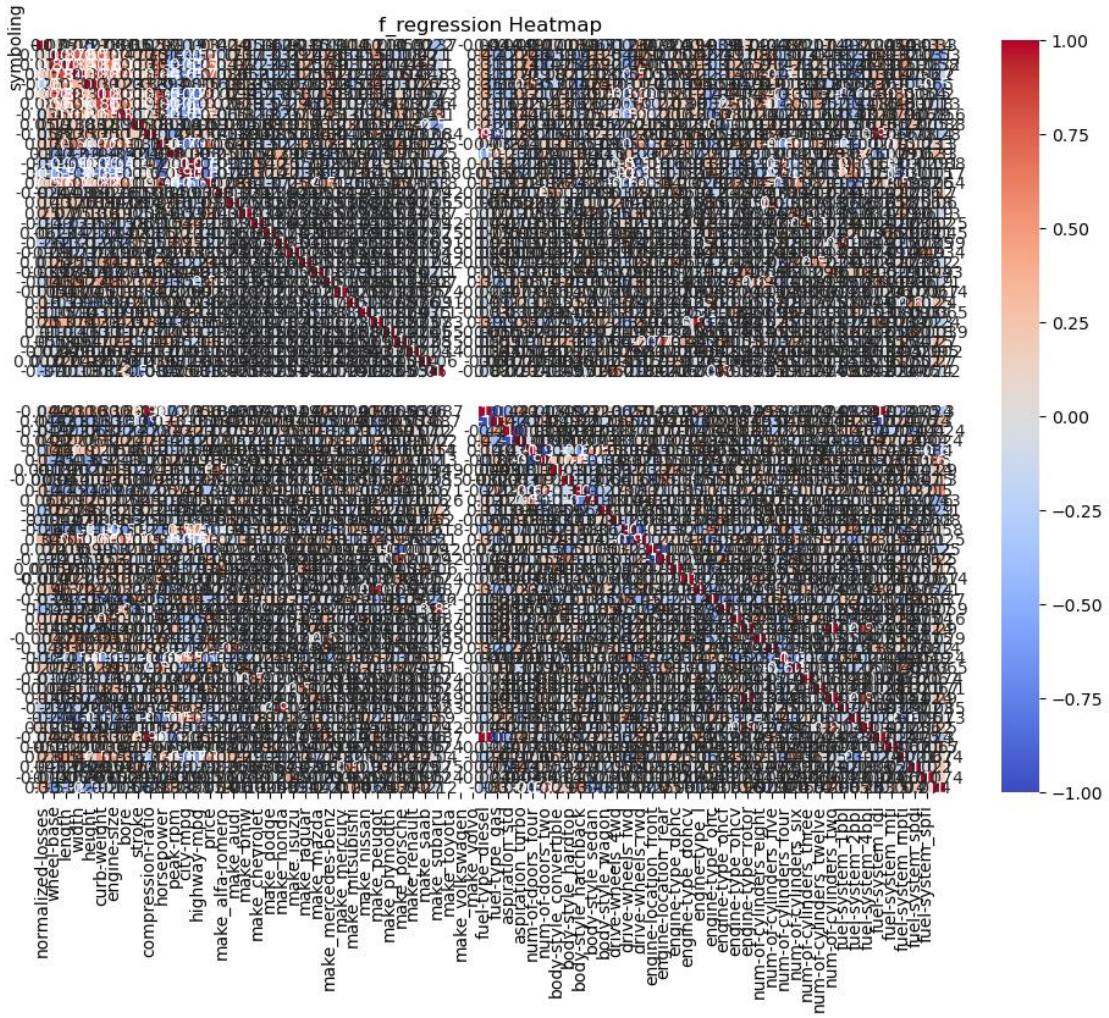
Heatmap для Distance Correlation:

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

# Визначте distance correlation між ознаками
distance_corr_matrix = np.zeros((train_set_scaled.shape[1], train_set_scaled.shape[1]))

for i in range(train_set_scaled.shape[1]):
    for j in range(train_set_scaled.shape[1]):
        distance_corr_matrix[i, j] = mic(train_set_scaled[:, i], train_set_scaled[:, j])

# Створіть heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(distance_corr_matrix, annot=True, cmap='coolwarm', xticklabels=localData_encoded.drop('symboling', axis=1).columns,
            yticklabels=localData_encoded.drop('symboling', axis=1).columns)
plt.title('Distance Correlation Heatmap')
plt.show()
```



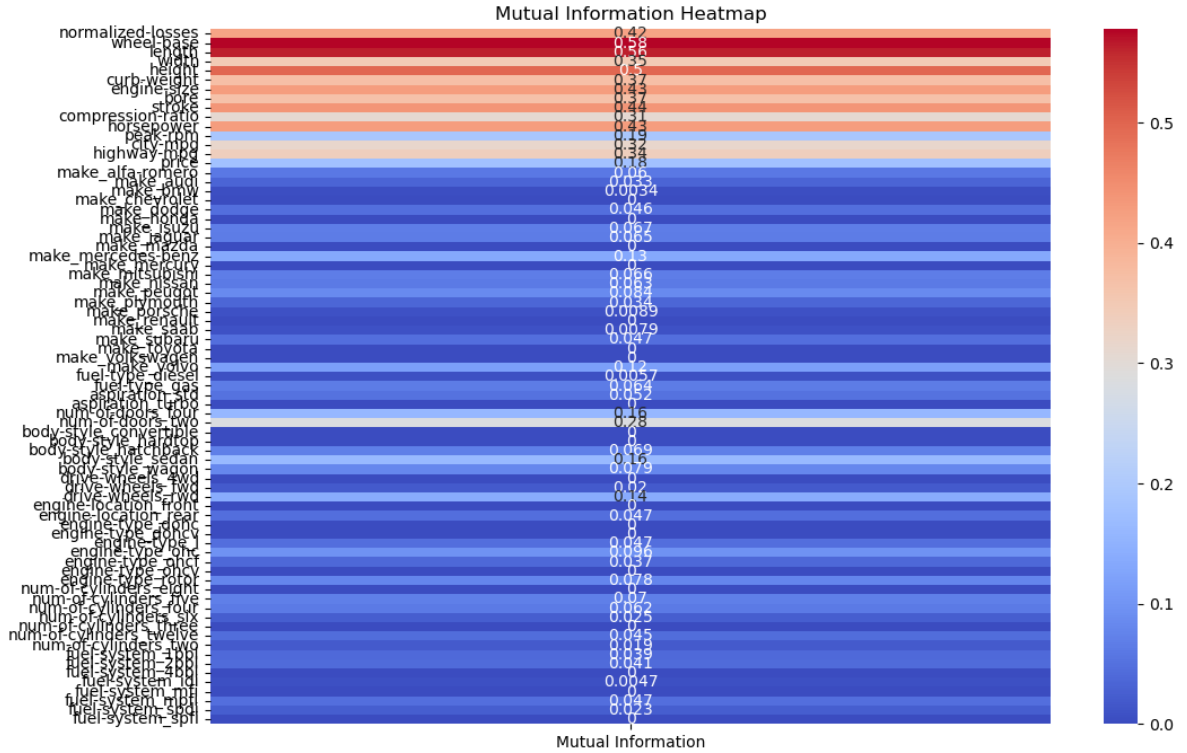
Heatmap для f_regression:

```
# Визначте f_regression між ознаками
f_regression_matrix = np.corrcoef(train_set_scaled.T, train_y)

# Створіть heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(f_regression_matrix, annot=True, cmap='coolwarm', xticklabels=localData_encoded.drop('symboling', axis=1).columns,
            yticklabels=['symboling'])
plt.title('f_regression Heatmap')
plt.show()
```



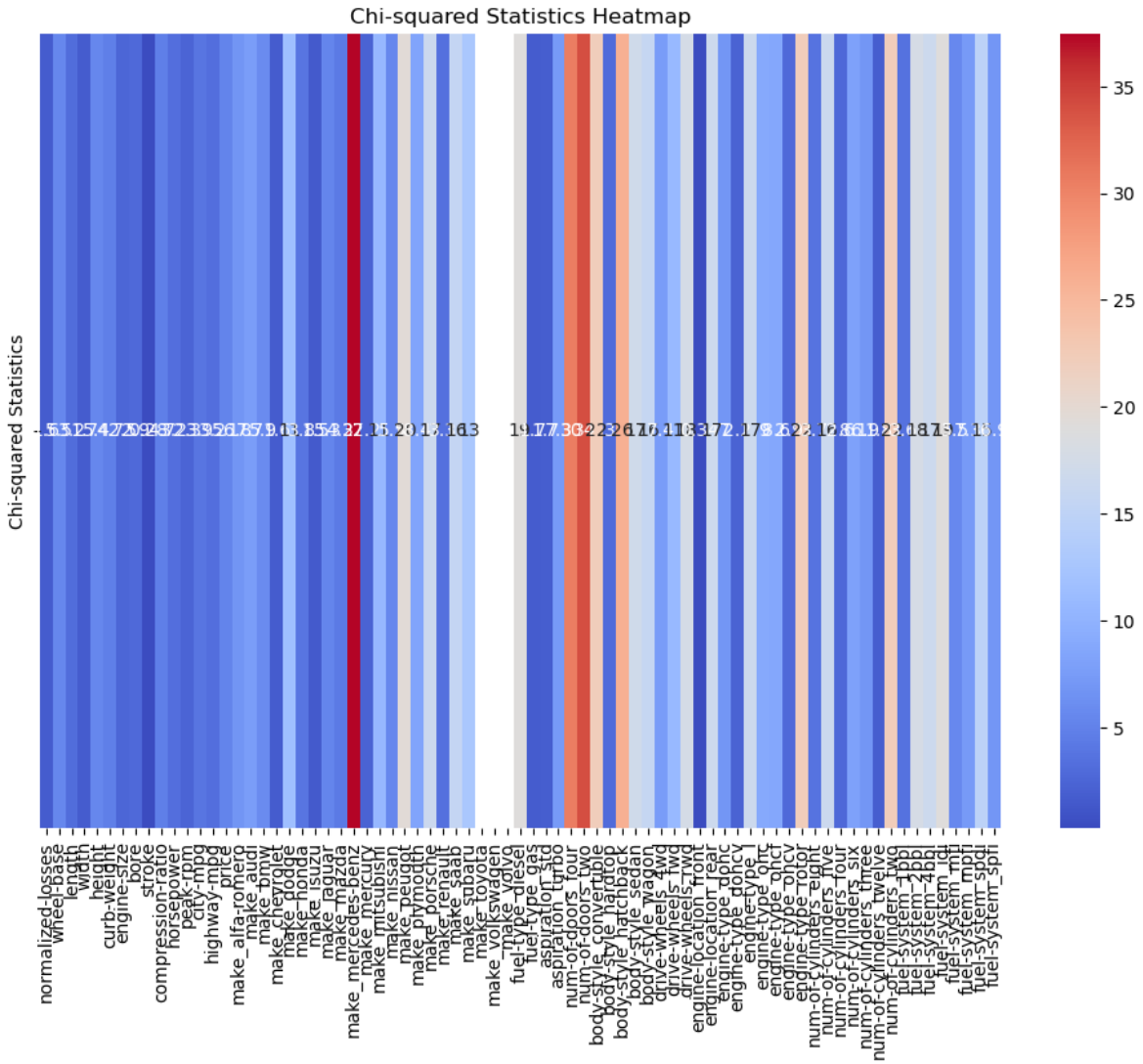
Heatmap для Mutual Information

```
# Визначте mutual information між ознаками
mutual_info_matrix = np.zeros((train_set_scaled.shape[1], 1))

for i in range(train_set_scaled.shape[1]):
    mutual_info_matrix[i, 0] = mutual_info_regression(train_set_scaled[:, i].reshape(-1, 1), train_y)

# Створіть heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(mutual_info_matrix, annot=True, cmap='coolwarm', xticklabels=['Mutual Information'],
            yticklabels=localData_encoded.drop('symboling', axis=1).columns)
plt.title('Mutual Information Heatmap')
plt.show()
```



Heatmap для Chi-squared Statistics

```
# Визначте chi2 між ознаками
chi2_matrix, _ = chi2(train_set_scaled, train_y)

# Створіть heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(chi2_matrix.reshape(1, -1), annot=True, cmap='coolwarm', xticklabels=localData_encoded.drop('symboling', axis=1).colu
            yticklabels=['Chi-squared Statistics'])
plt.title('Chi-squared Statistics Heatmap')
plt.show()
```



Отже,

1. **Взаємозалежність ознак:** З проведених методів вибірки ознак можна зрозуміти, які ознаки мають високу чи низьку взаємозалежність з цільовою змінною (symboling).

2. **Важливість ознак:** З ранжування ознак за різними методами виділення ознак можна визначити, які ознаки вносять значущий внесок у прогнозування цільової змінної.

3. **Диверсифікація результатів:** Використання різних методів дозволяє отримати різні погляди на взаємодію між ознаками та цільовою змінною. Це може бути корисним для впевненості у важливості певних ознак або для

ідентифікації аспектів, які можуть бути важливими в одних випадках та менш важливими в інших.

4. **Обробка пропущених значень:** Код включає обробку пропущених значень, використовуючи середнє значення відповідної колонки. Це може бути корисним для покращення аналізу та моделювання, оскільки багато алгоритмів машинного навчання вимагають повних даних.

4. Висновки до роботи

Під час виконання лабораторної роботи 3, я отримала практичні навички аналізу та вибору значущих ознак для моделі за допомогою кореляційного аналізу, таблиць сопряжіння, аналізу багатомірні залежності та дихотомії, дисперсійного аналіз – ANOVA, критерій Хі-квадрат тощо.