Лабораторна робота №5

Регресія. Прогноз ціни на будинки

Мета роботи: набути практичних навичок побудови лінійної регресії, опрацювання методу найменших квадратів, використання різних методів побудови моделі і її оцінки.

Література

sklearn.linear_model.LinearRegression: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html

Зміст роботи

Завдання 1. Створити модель для прогнозування ціни продажу будинків, беручі до уваги інформацію про особливості будинку. Для навчання моделі використати дані про житло США - USA_housing.csv.

У завданні буде використана модуль statsmodels Python для реалізації лінійної регресії методом звичайних найменших квадратів (OLS).

– Завдання рекомендується виконувати в Google Colaboratory.

– Бібліотеки, які знадобляться для роботи:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

– Завантажити файл USA housing.csv у фрейм дані з назвою housingDF;

– Продивитися перші записи;

– Продивитися назви всіх стовпців:

```
housingDF.columns.values
 array(['Avg. Area Income', 'Avg. Area House Age',
        'Avg. Area Number of Rooms', 'Avg. Area Number of Bedrooms',
        'Area Population', 'Price', 'Address'], dtype=object)
```

Особливості набору даних:

– *Avg. Area Income*: середній дохід жителів міста, в якому розташований будинок.
– *Avg. Area House Age*: середній вік будинків у тому самому місті.
– *Avg. Area Number of Rooms*: середня кількість кімнат будинку .
– *Avg. Area Number of Bedrooms*: середня кількість спалень будинку.

- *Area Population*: населення міста, в якому розташований будинок.
- *Price:* ціна, за якою будинок був/буде проданий.
- *Address:* адреса будинку.

– Продивитися розмір набору даних.

– Продивитися інформацію про дані.

– Зробити висновки.

Дослідити дані.

– *Парний графік* відображає парні зв'язки міжзмінними. Діагональні графіки показують однофакторний розподіл даних.

```
sns.pairplot(housingDF, kind='reg')
```

– *Діаграма розкиду.* Можна отримати розподіл лише для однієї змінної за допомогою функції seaborn distplot. Подивитися на розподіл цільової змінної, ціни.

```
sns.distplot(housingDF.Price)
```

– *Теплова карта.* Можна подивитися на кореляцію між функціями та цільовою змінною за допомогою теплової карти.

```
mask = np.zeros_like(housingDF.corr()*-1)
mask[np.triu_indices_from(mask)] = True

#fig,ax = plt.subplots(figsize=(12,12))
sns.heatmap(housingDF.corr(), mask=mask, annot=True)
```

– Зробити висновки.

– *Очищення даних.* Потрібно перевірити набір на відсутність. Можна перевірити це, побудувавши теплову карту нульових значень у наборі даних.

```
fig, ax = plt.subplots(figsize=(12,8))
sns.heatmap(housingDF.isnull(), cmap='coolwarm', yticklabels=Fals
e, cbar=False, ax=ax)
```

Як і очікувалося, у наборі даних немає червоних ліній, що вказують на NaN.

– *Колонка адреси.* Стовпець адреси має текстовий формат, тому його не можна додати до моделі scikit-learn. Цю функцію можна проаналізувати для створення нових цільових змінних, таких як тип вулиці тощо. Для цього прикладу можна просто видалити стовпець і подивитися, як працює модель з урахуванням решти функцій.

```
housingDF['Address'].head()
housingDF.drop('Address',axis=1,inplace=True)
```

Створення моделі лінійної регресії

Функція OLS() модуля statsmodels.api використовується для виконання регресії. Вона повертає об'єкт OLS. Потім для цього об'єкта викликається метод fit() для підгонки лінії регресії до даних.

– Створити модель *ціна ~ середній дохід*. Надрукувати коефіцієнти.

```
import statsmodels.formula.api as sf

model = sf.ols('Price ~ avgAreaIncome', housingDF.rename(index=st
r, columns={"Avg. Area Income":"avgAreaIncome"}))
fitted_model = model.fit()
print(fitted_model.params)
```

– Можна продивитися значення модельних коефіцієнтів з аргументу params:

```
b0 = fitted_model.params[0]
b1 = fitted_model.params[1]

print("y = {} + {}*(avg. area income)".format(round(b0,3),round(b1,3)))
```

– Знайти прогнозовану ціну будинку у районі з середнім доходом $100 000. Вона становить:

```
y = b0 + b1*100000
print ("Прогнозована ціна будинку для будинку в регіоні з середнім доходом
 $100 000 становить${}".format(round(y,2)))
```

```
Прогнозована ціна будинку для будинку в регіоні з середнім доходом $100 000 становить$1897968.84
```

Використання statsmodel для прогнозування

Цей прогноз можна зробити безпосередньо за допомогою статистичної моделі.

– Створити DataFrame для використання з інтерфейсом формул Statsmodels

```
avg_area_income = pd.DataFrame({'avgAreaIncome': [100000]})
```

– Продивитися записи;

```
avg_area_income.head()
```

– Використати створену вище модель для прогнозування ціни

```
sales = fitted_model.predict(pd.DataFrame({'avgAreaIncome':[100000]}))
print(sales)
```

```
0    1.897969e+06
dtype: float64
```

Як і очікувалося, модель статистики повернула той самий результат, що й лінійна функція вище.

Побудувати лінію тренду

– Створити DataFrame з мінімальними та максимальними значеннями змінної Avg. Area Income: середній дохід жителів міста, в якому розташований будинок.

```
avg_area_income = pd.DataFrame({'avgAreaIncome': [housingDF['Avg. Area Inc
ome'].min(), housingDF['Avg. Area Income'].max()]})
print(avg_area_income.head())
```

– Зробити прогнози для цих значень та зберегти його

```
price_predictions = fitted_model.predict(avg_area_income)
print(price_predictions)
```

– Побудувати дані спостережень

```
housingDF.plot(kind='scatter', x='Avg. Area Income', y='Price')
```

– Побудувати лінію тренду

```
plt.plot(avg_area_income, price_predictions, c='red', linewidth=2)
```

Наскільки добре модель відповідає даним?

– Метод summary () використовується для отримання таблиці, яка дає розгорнутий опис результатів регресії.

```
print(fitted_model .summary())
```

– Розрахувати значення R2 (R-squared) для простої моделі лінійної регресії можна наступним чином:

```
fitted_model.rsquared
```

Найпоширенішим способом оцінки загальної відповідності лінійної моделі наявним даним є обчислення значення R^2 *(коефіцієнт детермінації).* R^2 *приймає значення від 0 до 1, вищий показник вважається кращим.*

Зауважимо, що R^2 *не вказує на те, чи справді регресійна модель гарна. Можна мати низьке значення* R^2 *для хорошої моделі або високе значення для моделі, яка не відповідає даним!*

Завдання 2. Множинна лінійна регресія. Створити модель з усіма змінними набору даних USA_housing.csv.

– Перейменувати стовпці, оскільки OLS не приймає пробілів.

```
# Rename the columns since ols doesn't take spaces
tempDF = housingDF.rename(index=str,columns={
        "Avg. Area Income": "avgAreaIncome",
        "Avg. Area House Age": "avgAreaHouseAge",
        "Avg. Area Number of Rooms" : "avgAreaNumOfRooms",
        "Avg. Area Number of Bedrooms": "avgAreaNumOfBedrooms",
        "Area Population": "areaPopulation"
    })
```

– Створити модель.

```
multi_model = sf.ols(formula='Price ~ avgAreaIncome + avgAreaHouseAge + av
gAreaNumOfRooms + avgAreaNumOfBedrooms', data=tempDF)
fitted_multi_model = multi_model.fit()

# print the coefficients
print(fitted_multi_model.params)
```

– Роздрукувати короткий опис отриманої моделі.

– Визначте змінні які найкраще підходять і позитивно пов’язані з цільовою змінною (ціна).

– Включити в модель ті змінні які найкраще підходять

Завдання 3. Лінійна регресія у scikit-learn. Робота, яка виконана за допомогою Statsmodels, також може здійснюватися за допомогою scikit-learn.

– Для початку потрібно розділити дані на два масиви: (1) масив X із характеристиками та (2) масив з цільовою змінною (ціною).

```
X = housingDF.drop('Price',axis=1)
y = housingDF.Price
```

– Розділити дані на набір для навчання та тестування.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3, ra
ndom_state=101)
```

– Створити та навчити модель.

```
from sklearn.linear_model import LinearRegression

lm = LinearRegression()
lm.fit(X_train,y_train)
```

– Подивитися на параметри моделі.

```
print("Intercept: {}".format(lm.intercept_))
```

```
print("Coefficients: {}".format(lm.coef_))
```

– Поєднати назви змінних із коефіцієнтами для кращого сприйняття.

```
coeff_df = pd.DataFrame(lm.coef_,X.columns,columns=['Coefficient'])
coeff_df
```

	Coefficient
Avg. Area Income	21.617635
Avg. Area House Age	165221.119872
Avg. Area Number of Rooms	121405.376596
Avg. Area Number of Bedrooms	1318.718783
Area Population	15.225196

– Зробити прогноз. Спрогнозувати вартість житла з огляду на наступні особливості:

- *Avg. Area Income* = $65000
- *Avg. Area House Age* = 7
- *Avg. Area Number of Rooms* = 7
- *Avg. Area Number of Bedrooms* = 2
- *Area Population* = 35000

```
lm.predict([[65000, 7, 7, 2, 35000]])
```

– Обчислити R^2

```
lm.score(X_train, y_train)
```

– Зробити прогноз для тестового набору та подивимося, наскільки добре працює модель.

```
predictions = lm.predict(X_test)
plt.scatter(y_test, predictions)
```

– Зобразити похибку у вигляді гістограми.

```
sns.distplot((y_test-predictions),bins=50)
```

Альтернативні показники оцінювання

Метрики оцінки регресії:

***Середня абсолютна похибка** (MAE) — це середнє абсолютне значення похибок:*

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

MAE = середня абсолютна похибка
y_i = передбачення
x_i = справжнє значення
n = загальна кількість точок даних

Середня квадратична помилка *(MSE) – це середнє квадратичних помилок:*

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = середня квадратична помилка
n = кількість точок даних
Y_i = спостережувані значення
$\hat{Y}_i$ = прогнозовані значення

Середньоквадратична помилка *(RMSE) – це квадратний корінь із середньої квадратичної помилки.*

Вищезазначене вважається ***функціями втрат****, і мета полягає в тому, щоб мінімізувати її.*

```
#from sklearn.metrics import mean_squared_error
from sklearn import metrics

print('MAE:', metrics.mean_absolute_error(y_test, predictions))
print('MSE:', metrics.mean_squared_error(y_test, predictions))
print('RMSE:', np.sqrt(metrics.mean_squared_error(y_test, predictions)))
```

Контрольні запитання

1. Яке основне призначення регресійного аналізу?
2. Які типи регресійних моделей виділяють?
3. Який вигляд має лінійна регресійна модель?
4. Які висуваються основні вимоги до специфікації моделі та відомих експериментальних даних?
5. В чому полягає сутність методу найменших квадратів?
6. Як оцінити якість отриманої моделі?