

## Лабораторна робота 2: основні поняття

В даній роботі ми будемо працювати з даними про автомобілі, продані на аукціоні ( Carvana Car Prediction ). Цільовими ознаками тут є наявність прихованих продавцем істотних недоліків.

Carvana — компанія-перекупц автомобілів, відповідно, її задача — не допустити скупку автомобілів з прихованими недоліками, тому що в майбутньому їх важче продати.

### 1

Завантажте дані про продані на аукціоні автомобілі. Перегляньте "сирі" дані.

Перегляньте файл DataDictionary-ua.txt та зрозумійте, що означають стовпчики матриці. Виведіть статистику за стовпчиками.

In [82]:

```
import pandas as pd
import numpy as np
```

In [3]:

```
df = pd.read_csv("data.csv")
len(df[df['IsBadBuy']==0])
```

Out[3]:

64007

### 2

Побудуйте графік розсіювання з пробігом в якості вісі абсцис та ціною

MMRCurrentRetailAveragePrice в якості вісі ординат. Автомобілі без недоліків відмітьте зеленим кольором, з прихованими недоліками - червоним.

In [5]:

```
%matplotlib inline
import matplotlib.pyplot as plt
```

In [6]:

```
temp = df["IsBadBuy"].dropna()

color = ['Green' if temp == 1 else 'Red' for temp in df["IsBadBuy"]]

plt.scatter(df["VehOdo"], df["MMRCurrentRetailAveragePrice"], c = color)

plt.xlabel('Пробіг')
plt.ylabel('Ціна')
plt.title('Графік розсіювання автомобілів в залежності від пробігу')
```

Out[6]:

Text(0.5, 1.0, 'Графік розсіювання автомобілів в залежності від пробігу')



## Регресія

### 3а

Імпортуйте із бібліотеки `sklearn` всі моделі машинного навчання:

```
from sklearn import *
```

Трактуючи задачу про прогнозування наявності недоліків як задачі регресії, натренуйте лінійну модель [LinearRegression](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html) ([https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LinearRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)) на якійсь підмножині ознак (наприклад, тих самих `VehOdo` и `MMRCurrentRetailAveragePrice`).

З якою проблемою ви затовхнулись? Яку способи її рішення ви можете запропонувати?

Функції, що можуть бути корисними при рішенні: `.fit()`, `.loc[]`, `pd.notnull()`

Створення моделі: `model1 = linear_model.LinearRegression()`

Тренування: `model1.fit(x, y)`

In [85]:

```
from sklearn.linear_model import LinearRegression

selected_features = df[['VehOdo', 'MMRCurrentRetailAveragePrice', 'IsBadBuy']].dropna()
X = selected_features.drop(['VehOdo', 'MMRCurrentRetailAveragePrice'], axis=1)

##Так як данні параметру 'IsBadBuy' однорідні, далі ми виконаємо рандомізацію данного стовпця
randomized_row = np.random.choice([0, 1], size=len(selected_features), replace=True, p=[0.5, 0.5])
selected_features['IsBadBuy'] = randomized_row

y = selected_features['IsBadBuy']

model = LinearRegression()
model.fit(X, y)
coefficients = model.coef_
intercept = model.intercept_

print("Coefficients:", coefficients)
print("Intercept:", intercept)
```

```
Coefficients: [-6.67429946e-05]
Intercept: 0.5016320381645849
```

## 3b

Виконайте прогнозування для всіх об'єктів навчальної вибірки та привласніть результат змінній `prediction`

Функції, що можуть бути корисні при рішенні: `model.predict()`

In [86]:

```
predicted_x_val = model.predict(X)
predicted_x_val
```

Out[86]:

```
array([0.50163204, 0.50163204, 0.50163204, ..., 0.50163204, 0.50163204,
       0.50163204])
```

### 3c

Перетворіть отриманий вектор прогнозувань `prediction` до значень `{0,1}`. Це можна виконати, наприклад, використовуючи list comprehensions: <https://docs.python.org/3/tutorial/datastructures.html#list-comprehensions> (<https://docs.python.org/3/tutorial/datastructures.html#list-comprehensions>).

```
predictionClass = [1 if prediction[i] > 0.5 else 0 for i in
range(prediction.shape[0])]
```

In [69]:

```
predictionClass = [1 if value > 0.5 else 0 for value in predicted_x_val]
predictionClass
```

```
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
```

### 3d

Побудуйте звіт з якості класифікації та матриці похибок. Як зміняться звіт та матриця похибок, якщо зміниться поріг в завданні 3c (за замовчуванням його значення рівне 0.5)?

Функції, що можуть бути корисні при рішенні: `print(metrics.classification_report(...))`, `print(metrics.confusion_matrix(...))`

Функція `classification_report` для кожного класу об'єктів рахує точність (`precision`) в цьому класі та повноти (`recall`). Повнота - це процент об'єктів даного класу, які ваш метод прогнозування також віднести до даного класу, серед всіх об'єктів даного класу. Точність (`precision`) - це те саме, тільки серед всіх об'єктів, спрогнозованих для даного класу.

Функція `confusion_matrix` повертає матрицю з кількістю об'єктів. Номера стовпчиків матриці - це номер спрогнозованих класів, рядки - це номери правильних класів. Наприклад, елемент `M[0,1]` - це кількість машин, де насправді `IsBadBuy = 0`, а ви спрогнозували 1.

In [70]:

```
from sklearn import *
print(metrics.classification_report(y, predictionClass))
print(metrics.confusion_matrix(y, predictionClass))
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.00      | 0.00   | 0.00     | 36286   |
| 1            | 0.50      | 1.00   | 0.67     | 36382   |
| accuracy     |           |        | 0.50     | 72668   |
| macro avg    | 0.25      | 0.50   | 0.33     | 72668   |
| weighted avg | 0.25      | 0.50   | 0.33     | 72668   |

```
[[ 0 36286]
 [ 0 36382]]
```

C:\Users\maxim\anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

C:\Users\maxim\anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

C:\Users\maxim\anaconda3\lib\site-packages\sklearn\metrics\\_classification.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined and being set to 0.0 in labels with no predicted samples. Use `zero\_division` parameter to control this behavior.

```
_warn_prf(average, modifier, msg_start, len(result))
```

## Класифікація

### 4a

Трактуючи задачу як за задачу класифікації, побудуйте модель класифікації, побудуйте модель класифікації «решаюче дерево» глибиною 20 (все аналогічно лінійної регресії).

Функції, що можуть бути корисні при рішенні: `tree.DecisionTreeClassifier(max_depth=20)`

In [71]:

```
tree_model = tree.DecisionTreeClassifier(max_depth=20)
tree_model.fit(X, y)
```

Out[71]:

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=20)
```

## 4b

Решаюче дерево прогнозує ймовірності. За допомогою побудованої моделі розрахуйте ймовірності наявності недоліків, що приховуються.

Функції, що можуть бути корисні при вирішенні: `model.predict_proba()`

In [72]:

```
tree_predicted_val = model.predict(X)
tree_predicted_val
```

Out[72]:

```
array([0.50032955, 0.50032955, 0.50032955, ..., 0.50032955, 0.50032955,
       0.50032955])
```

## 4c

Перетворіть отриманий вектор прогнозувань `prediction` до значень `{0,1}`. Побудуйте звіт про класифікацію та матрицю похибок. Який метод виявився краще?

In [73]:

```
predictionClass = [1 if value > 0.5 else 0 for value in predicted_x_val]
predictionClass
```

```
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1,
1
```

## Крос-валідація

### 5a

Розділіть початкову вибірку на дві частини, наприклад, використовуючи [slices](https://pythonz.net/references/named/slice/) (<https://pythonz.net/references/named/slice/>):

```
dataTrain = data.loc[0:34999,] dataTest = data.loc[35000:69999,]
```

In [74]:

```
Test_data = X.loc[0:34999,]
Test_labels = y.loc[0:34999,]
Train_data = X.loc[35000:69999,]
Train_labels = y.loc[35000:69999,]

tree_model2 = tree.DecisionTreeClassifier(max_depth=20)
```

## 5b

На тренуйте рішенняе дерево на `dataTrain` та застосуйте до `dataTrain` та `dataTest`, підрахувавши для кожного випадку точність прогнозування.

Проведіть декілька експериментів для різних глибин дерев. Зазначте в коментарях, для яких глибин модель недонавчена, для яких перенавчена та де знаходиться точка раннього останову (зупинки).

Функції, що можуть бути корисними при вирішенні: `metrics.accuracy_score()`

In [75]:

```
tree_model2.fit(Train_data, Train_labels)
train_predictions = tree_model2.predict(Train_data)
test_predictions = tree_model2.predict(Test_data)

train_accuracy = metrics.accuracy_score(Train_labels, train_predictions)
test_accuracy = metrics.accuracy_score(Test_labels, test_predictions)

print(f'Точність на тренувальних даних: {train_accuracy}')
print(f'Точність на тестових даних: {test_accuracy}')
```

Точність на тренувальних даних: 0.5013491015557724  
Точність на тестових даних: 0.49723154603092634

# Рішаюча функція

## 6a

Повернемось до моделі рішенняого дерева глибиною 20, що побудовано за навчальною вибіркою `data`. Нехай ціна похибки невірного прогнозування 0 дорівнює 1000, а невірний прогноз 1 — 100.

Підрахуйте функцію втрат — середню похибку по всій навчальній вибірці.

In [76]:

```
# Припустимо, що ціни помилок вже визначені
price_of_error_0 = 1000
price_of_error_1 = 100

losses = []
for i in range(len(Train_labels)):
    true_label = Train_labels.iloc[i]
    predicted_label = train_predictions[i]

    if true_label == 0 and predicted_label == 1:
        loss += price_of_error_0
    elif true_label == 1 and predicted_label == 0:
        loss = price_of_error_1
    else:
        loss = 0

    losses.append(loss)

average_loss = sum(losses) / len(losses)

print(f'Середня похибка на навчальній вибірці: {average_loss}')
```

Середня похибка на навчальній вибірці: 49.865089844422755

## 6b

З'ясуйте, як потрібно змінити рішачу функцію `[0 if predictionProb[i][0] > 0.5 else 1 for i in range(prediction.shape[0])]`, щоб функція втрат була мінімальною?

Знайдіть оптимальну рішачу функцію та мінімальне значення функціоналу втрат методом підбору.



In [77]:

```
import numpy as np
#Ми будемо шукати оптимальні значення використовуючи оптимальний поріг на основі ймовірності

for threshold in np.arange(0.0, 1.01, 0.01):
    threshold_predictions = [0 if prob > threshold else 1 for prob in train_predictions]

    losses = []
    for i in range(len(Train_labels)):
        true_label = Train_labels.iloc[i]
        predicted_label = train_predictions[i]

        if true_label == 0 and predicted_label == 1:
            loss += price_of_error_0
        elif true_label == 1 and predicted_label == 0:
            loss = price_of_error_1
        else:
            loss = 0

    losses.append(loss)

average_loss = sum(losses) / len(losses)

print(f'Середня похибка на навчальній вибірці: {average_loss}')
```

Середня похибка на навчальній вибірці: 49.865089844422755

In [ ]: