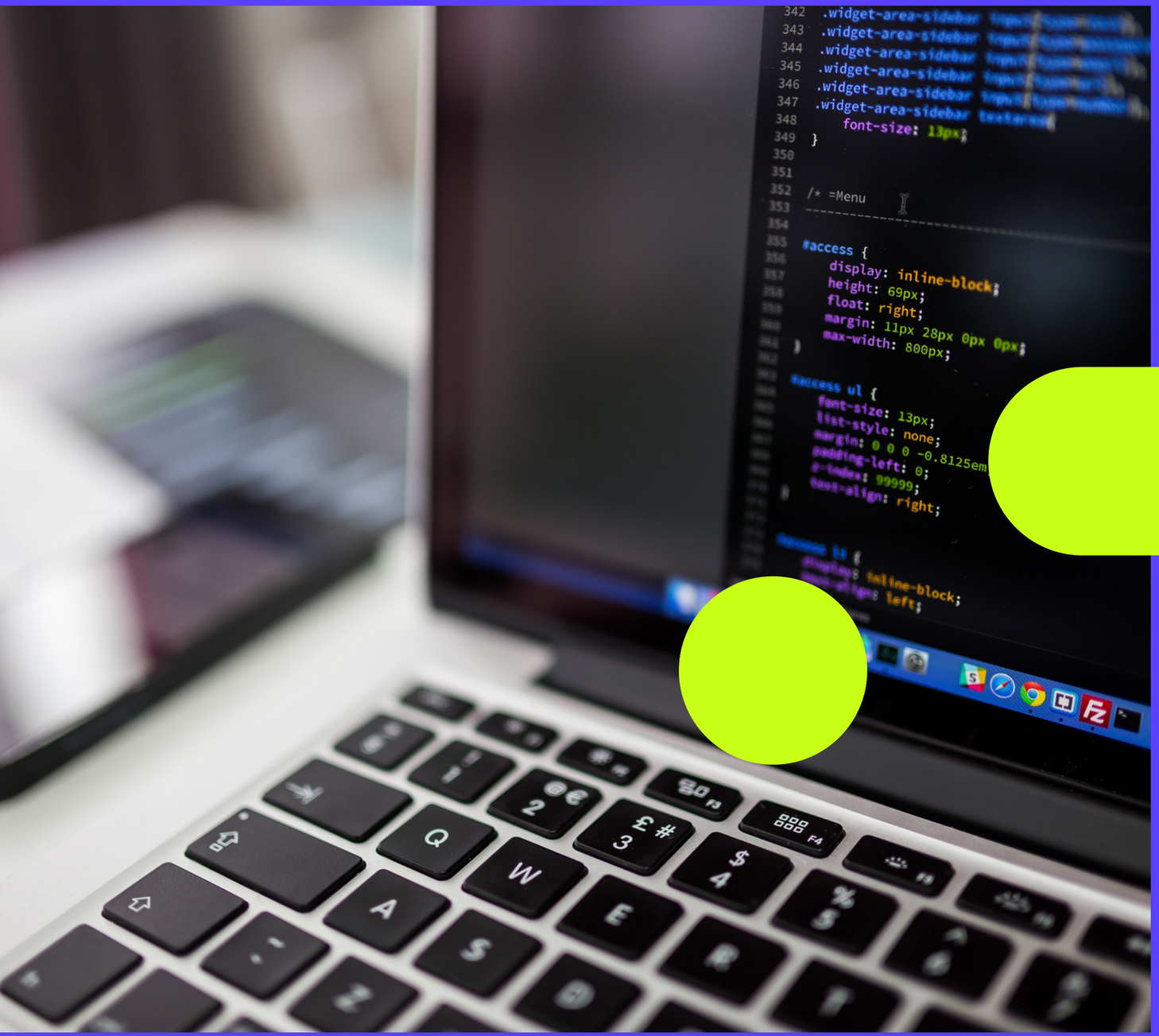


Машинное обучение

Проект, который благодаря машинному обучению, предсказывает купит ли человек курс

Наши цели:

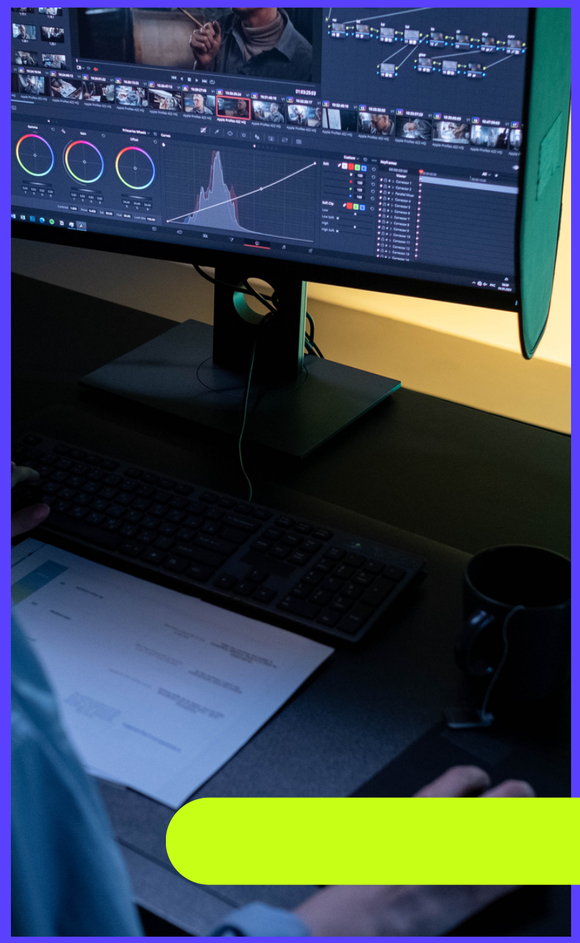
Научиться создавать математическую модель

Рассказать и показать код

Придумать гипотезу

Подвести итоги

ML – это...



Что такое машинное обучение?

Машинное обучение (ML) — это направление искусственного интеллекта, сосредоточенное на создании систем, которые обучаются и развиваются на основе получаемых ими данных.

Задача –
Определить по набору информации о пользователе его вероятность покупки курса

Начнем!

Гипотеза:

люди, учащиеся в университете, с наибольшей вероятностью купят курс

Для этого нам нужно:

1. Очистить данные для работы с моделью
2. Удалить ненужные данных
3. Создание математической модели

Ход работы

1. Подготовка: очистка данных для работы с моделью

```python
# ГИПОТЕЗА: люди, учащиеся в универе, с наибольшей вероятностью купят курс
#Импорт нужных библиотек
import pandas as pd
import matplotlib.pyplot as plt
df = pd.read_csv('train.csv')

#Очистка данных
def sex_apply(sex):
    if sex == 2:
        return 0 #мужчины
    return 1 #женщины
df['sex'] = df['sex'].apply(sex_apply)


def edu_status_apply(edu_status):
    if edu_status == "Undergraduate applicant":
        return 0
    elif (edu_status == "Student (Master's) "
        or edu_status == "Student (Bachelor's)"
        or edu_status == "Student (Specialist)"):
        return 1 #учащиеся
    elif  (edu_status == "Alumnus (Master's)"
        or  edu_status == "Alumnus (Bachelor's)"
        or edu_status == "Alumnus (Specialist)"):
        return 2 #окончившие
    else:
        return 3
df['education_status'] = df['education_status'].apply(edu_status_apply)
```

```python
def  occu_type_apply(occupation_type):
    if occupation_type == "university":
        return 1
    else:
        return 0
df['occupation_type'] = df['occupation_type'].apply(occu_type_apply)


def langsCleaner(lang):
    return lang.count(';')+1
df['langs'] = df['langs'].apply(langsCleaner)
```

Ход работы

2 Очистка: удаление ненужных данных

```
43  #Удаление ненужных данных
44  df.drop(['id', 'has_photo', 'has_mobile', 'followers_count', 'graduation', 'education_form',
45  'relation', 'last_seen', 'occupation_name', 'life_main', 'people_main', 'city', 'bdate',
46  'career_start', 'career_end'], axis = 1, inplace = True)
47
```

```
48  #Создание модели
49
50  from sklearn.model_selection import train_test_split
51  from sklearn.preprocessing import StandardScaler
52  from sklearn.neighbors import KNeighborsClassifier
53  from sklearn.metrics import accuracy_score, confusion_matrix
54
55
56  x = df.drop('result', axis = 1)
57  y = df['result']
58
59  train_x, test_x, train_y, test_y = train_test_split(x, y, test_size = 0.25)
60
61  sc = StandardScaler()
62  train_x = sc.fit_transform(train_x)
63  test_x = sc.transform(test_x)
64
65  classifier = KNeighborsClassifier(n_neighbors = 3)
66
67  classifier.fit(train_x, train_y)
68
69  pred_y = classifier.predict(test_x)
70  print('Точность теста:', round(accuracy_score(test_y, pred_y)*100, 2))
```

Ход работы

3 Создание математической модели

Итог:

1. Запускаем код несколько раз

2. После 3 запуска имеем точность свыше 80%!

Конец!

Спасибо за ваше внимание и время =)