Математическая модель анализирующая зависимость возможности покупки курса на основе данных о человеке

Презентация Дастана Мусрепова, о математической модели анализирующей зависимость возможности покупки курса на основе данных о человеке.

Мой код - Начало

```
import pandas
df = pandas.read_csv('train.csv')
df.drop(['id','bdate','has_photo','has_mobile','followers_count',
         'graduation','relation','life_main','people_main','city',
         'last_seen','occupation_name','career_start','career_end'],
         axis=1,inplace=True)
```

В этой части кода я импортирую библиотеку, читаю файл, и избавляюсь от ненужных столбцов. Нужными я посчитал пол, статус обучения, ‘langs’, ‘occupation_type’, форму обучения и результат.

Условия для данных для математической модели

1. Они должны быть числами

   Для этого в статус обучения мы пишем 1, 2, 3 и 4 в зависимости от того что стояло там до этого, а ‘langs’ и ‘occupation type’ меняем на 0 и 1.

2. Они не должны быть сравнимы когда это не надо

   Для этого столбец с полом мы меняем с 1 и 2 на 0 и 1, а столбец с формой обучения делим на несколько с помощью функций ‘fillna’.

Столбец с результатом состоит из 0 и 1. Он нам подходит.

Мой код - приведение к подлежащему виду

```
def sex_apply(sex):
    if sex==2:
        return 0
    return 1
df['sex']=df['sex'].apply(sex_apply)
```

```
def es_apply(es):
    if es=='Undergraduate applicant':
        return 1
    elif es.find('Student')!=-1:
        return 2
    elif es.find('Alumnus')!=-1:
        return 3
    else:
        return 4
df['education status']=df['education status'
                          ].apply(es_apply)
```

```
def langs_apply(langs):
    if langs.find('Русский')!=-1:
        return 0
    return 1
df['langs']=df['langs'].apply(langs_apply)
```

```
df['occupation_type'].fillna('university',
                             inplace=True)
def ot_apply(ot):
    if ot=='university':
        return 0
    return 1
df['occupation_type']=df['occupation_type'].apply(
    ot_apply)
```

```
df['education_form'].fillna('Full-time',
                            inplace=True)
df[list(pandas.get_dummies(df['education_form']
                           ).columns)
                           ]=pandas.get_dummies(
                               df['education_form'])
df.drop(['education_form'],axis=1,
        inplace=True)
```

Мой код - подготовка данных математической модели

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
```

Импортируем нужные билиотеки sklearn - а.

```
x=df.drop('result',axis=1)
y=df['result']
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.40)
```

Строим график и 40 % данных отдаем на тест.

Мой код - математическая модель и процент правильно предсказанных исходов

```
sc=StandardScaler()
x_train=sc.fit_transform(x_train)
x_test=sc.transform(x_test)

classifier=KNeighborsClassifier(n_neighbors=5)
classifier.fit(x_train,y_train)

y_pred=classifier.predict(x_test)
print('Процент правильно предсказанных исходов:',round(accuracy_score(y_test,y_pred)*100,2))
```

Делаем что - то сложное и печатаем процент правильно предсказанных исходов.

Процент правильно предсказанных исходов ~ 83 %.