МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Лабораторные работы по дисциплине

«Введение в машинное обучение»

Выполнил:

Студент гр.09-305

Шарафеев Марат

Преподаватель:

Шустова Е.П.

Казань-2024

**Задание №5. Логистическая регрессия.**

**Построение и применение однофакторной логистической регрессионной модели**

Найдите реальные данные для задачи однофакторной бинарной классификации и выполните для них следующее:

- Найдите коэффициенты логистической регрессионной модели.

- постройте логистическую кривую и облако точек.

- выполните прогноз переменной отклика для нескольких новых объектов и нескольких объектов обучающей выборки.

- проиллюстрируйте на графике результаты прогнозирования.

- сделайте выводы о качестве логистической регрессионной модели.

- результаты оформите в виде Word-файла и прикрепите его здесь (весь Python-код поместите в конец файла как приложение).

**Ход работы:**

1. Данные содержат анонимную информацию, такую как возраст, профессия, образование, рабочий класс и т.д. Цель состоит в том, чтобы обучить бинарный классификатор предсказывать доход, который имеет два возможных значения ">50 тыс." и "<50 тыс.". В наборе данных содержится 48842 экземпляра и 14 атрибутов. Данные содержат хорошее сочетание категориальных, числовых и отсутствующих значений. Данный датасет был взять из [Census Income Data Set](https://archive.ics.uci.edu/ml/datasets/census+income).

Зависимые переменные включают:

* age – возраст
* workclass - рабочий класс:
  + Private
  + Self-emp-not-inc
  + Self-emp-inc
  + Federal-gov
  + Local-gov
  + State-gov
  + Without-pay
  + Never-worked
* fnlwgt - вес
* education – образование
* education-num – численное значение колонки education
* marital-status – семейное положение
* occupation – профессия
* relationship – отношения
* race - раса
* sex - пол
* capital-gain – прирост капитала, отличных от заработной платы
* capital-loss – потеря капитала, отличных от заработной платы
* hours-per-week – часов в неделю
* native-country – родная страна

1. Подключим необходимые библиотеки и считаем данные:

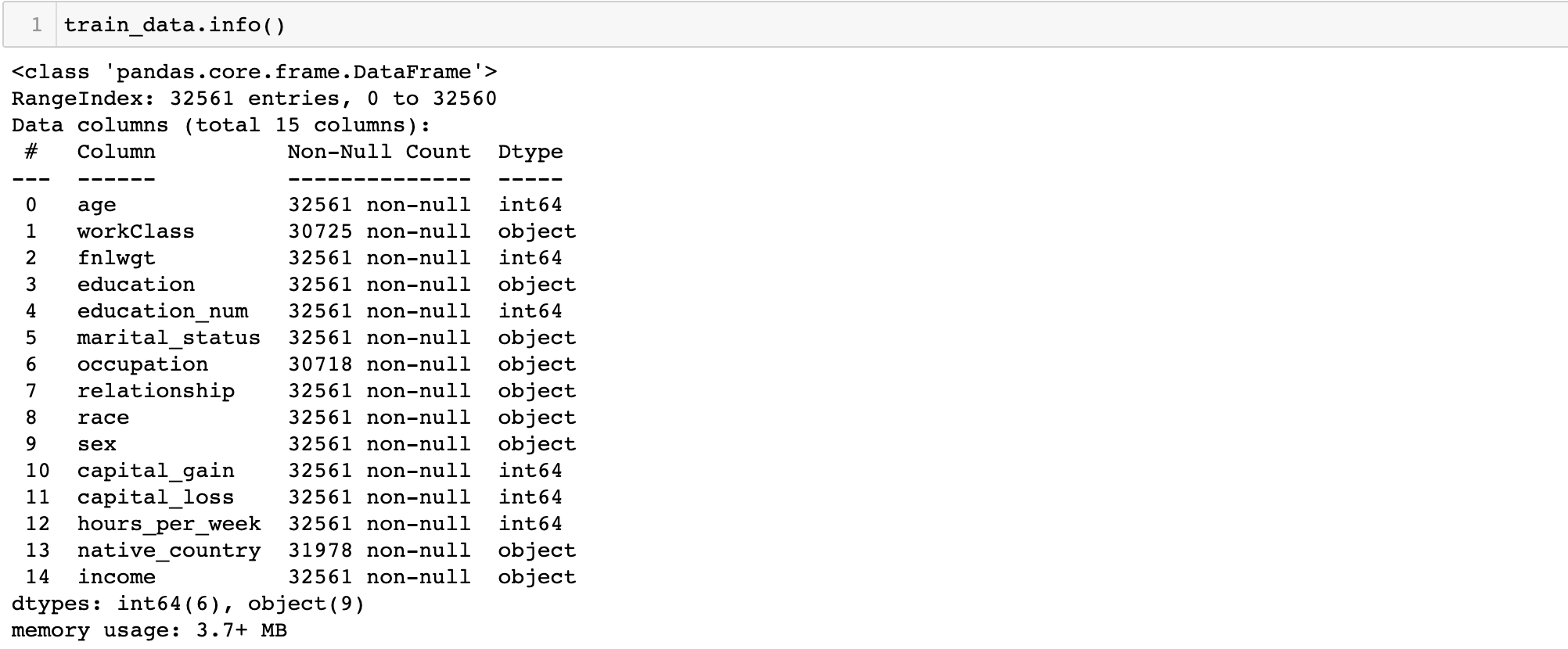


Выведем получившиеся данные:



В данных были некоторые пробелы до и после значений данных. Чтобы обрезать все пробелы, мы используем разделитель ‘ \*, \*’. В тестовом наборе данных есть странная первая строка, поэтому мы пропускаем строку, используя skiprows=1. Отсутствующие значения в наборе данных обозначены '?' Далее мы изучим данные. Это важный шаг перед тем, как приступить к построению модели.

1. Изучим внимательней наши данные:



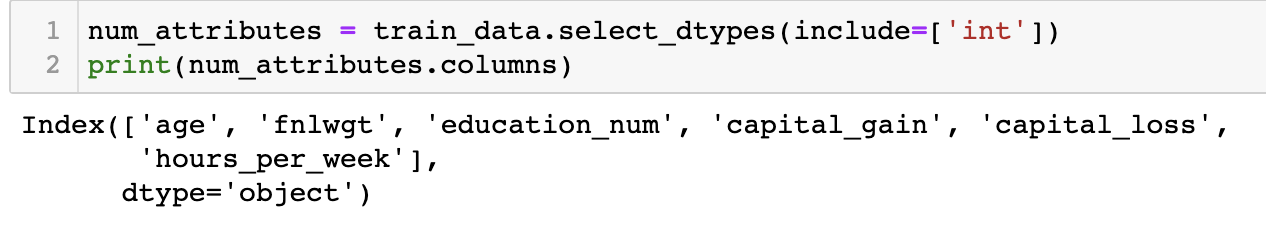
Наблюдение:

* В обучающем наборе данных 32561 выборка .
* В наборе данных есть как категориальные, так и числовые столбцы.
* В столбцах workClass , occupation , native-country пропущены значения

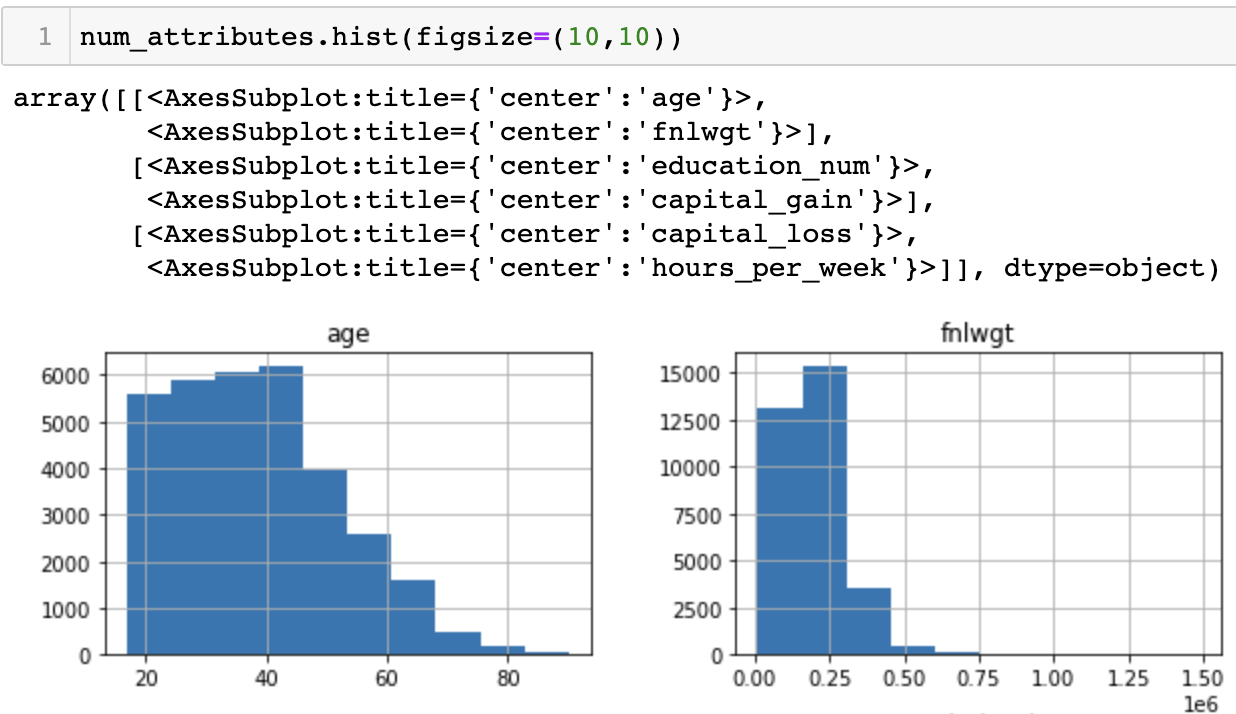
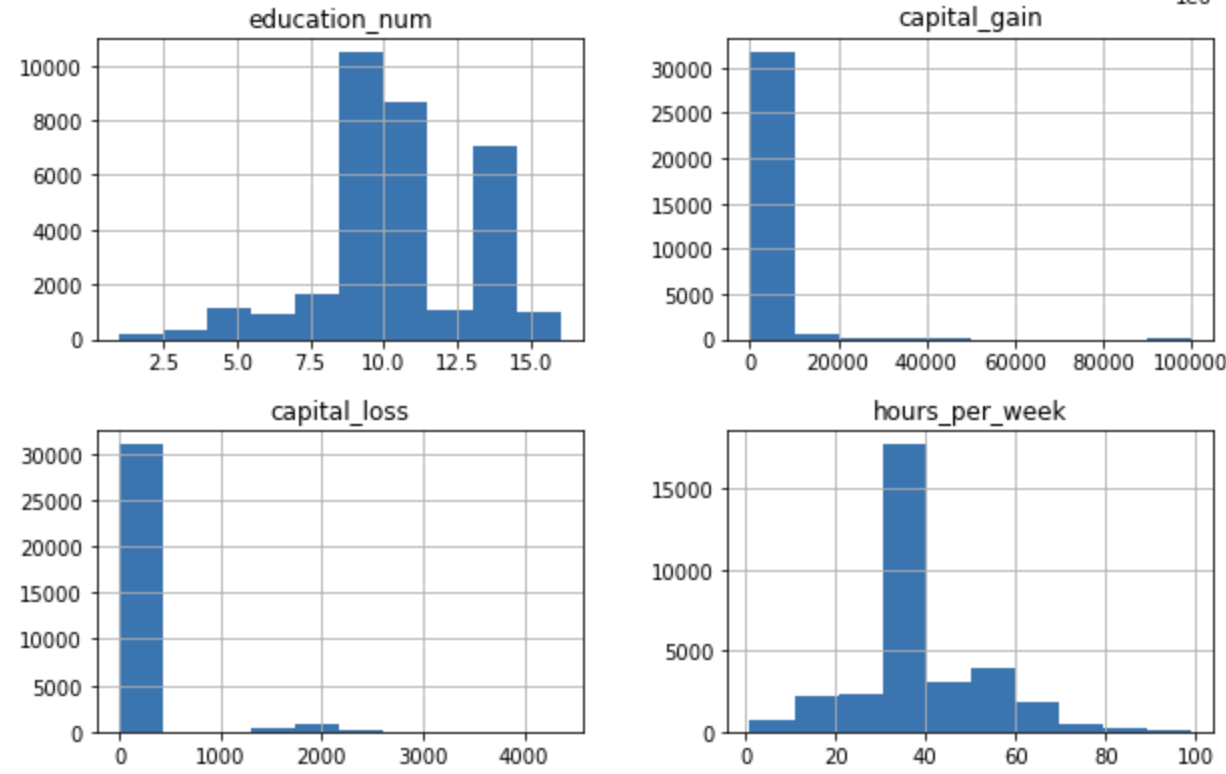
Аналогично, для тестового набора данных

* Есть 16281 образец
* Нет пропущенных значений

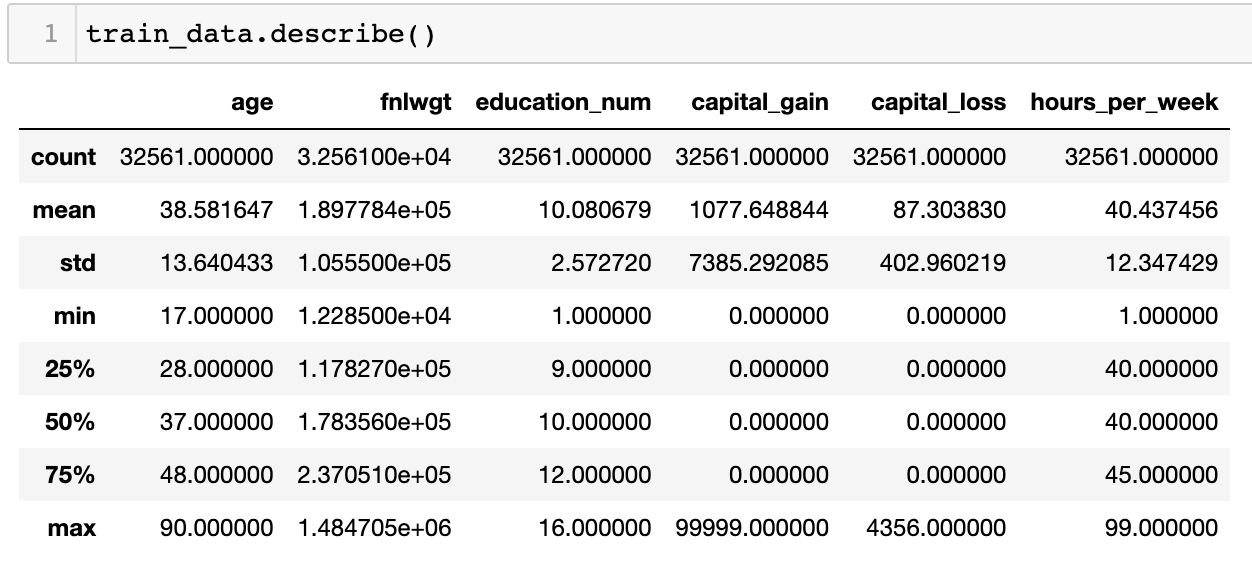
Обработаем числовые столбцы. Используем select\_dtypes для отборки числовых столбцов.



1. Давайте посмотрим числовые и категориальные данные с помощью некоторых визуализаций.

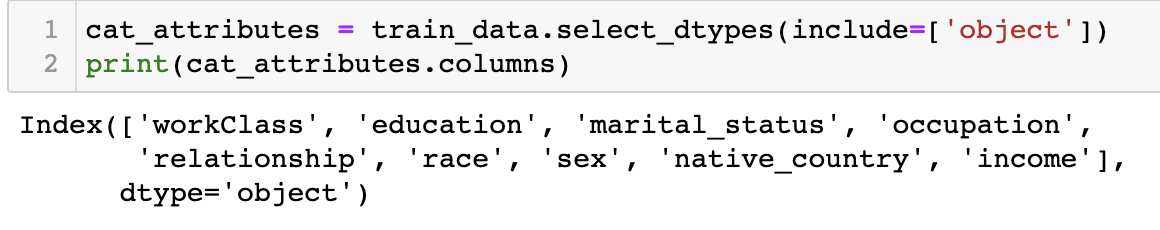
 

1. Более подробную информацию о данных можно получить, используя train\_data.describe().

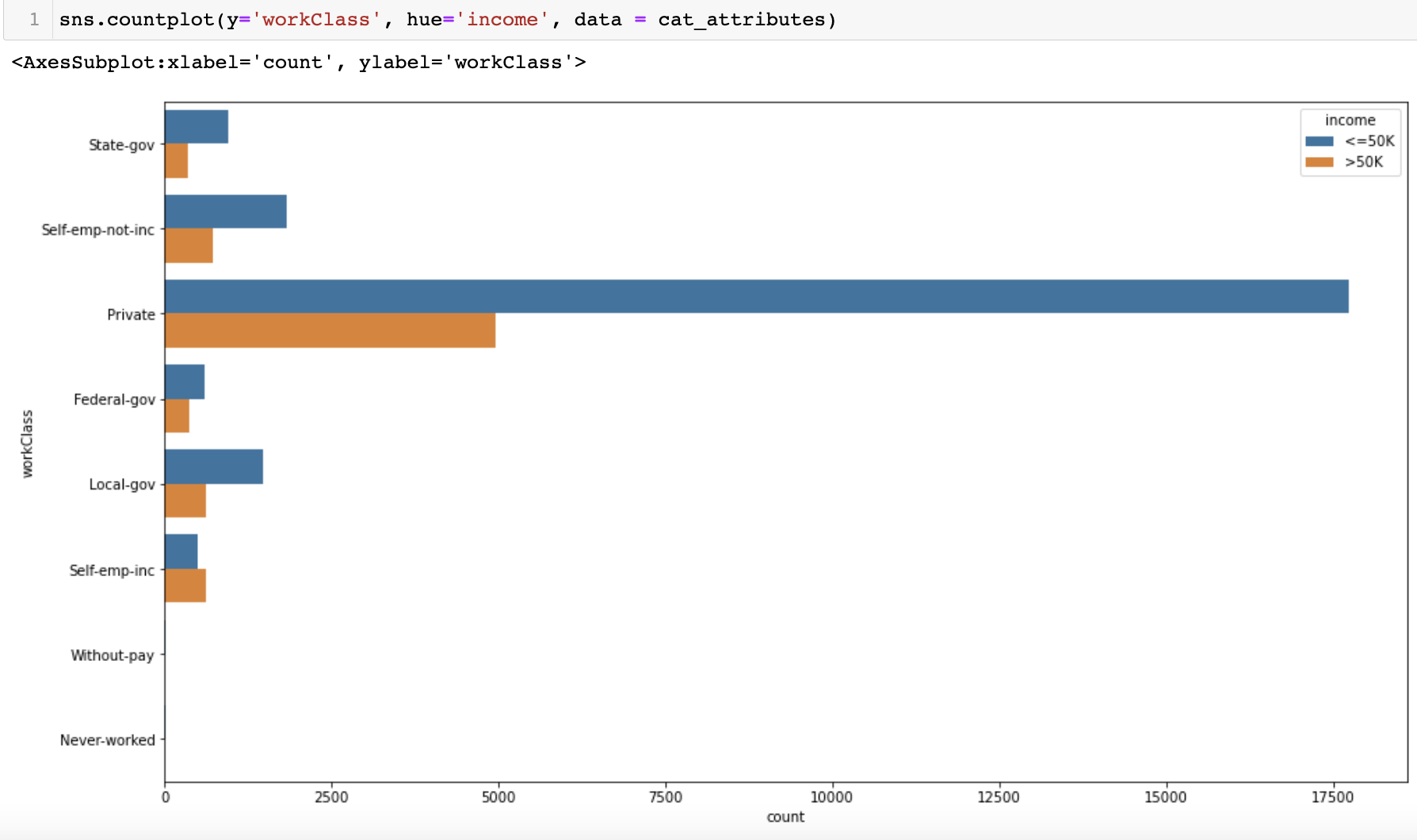


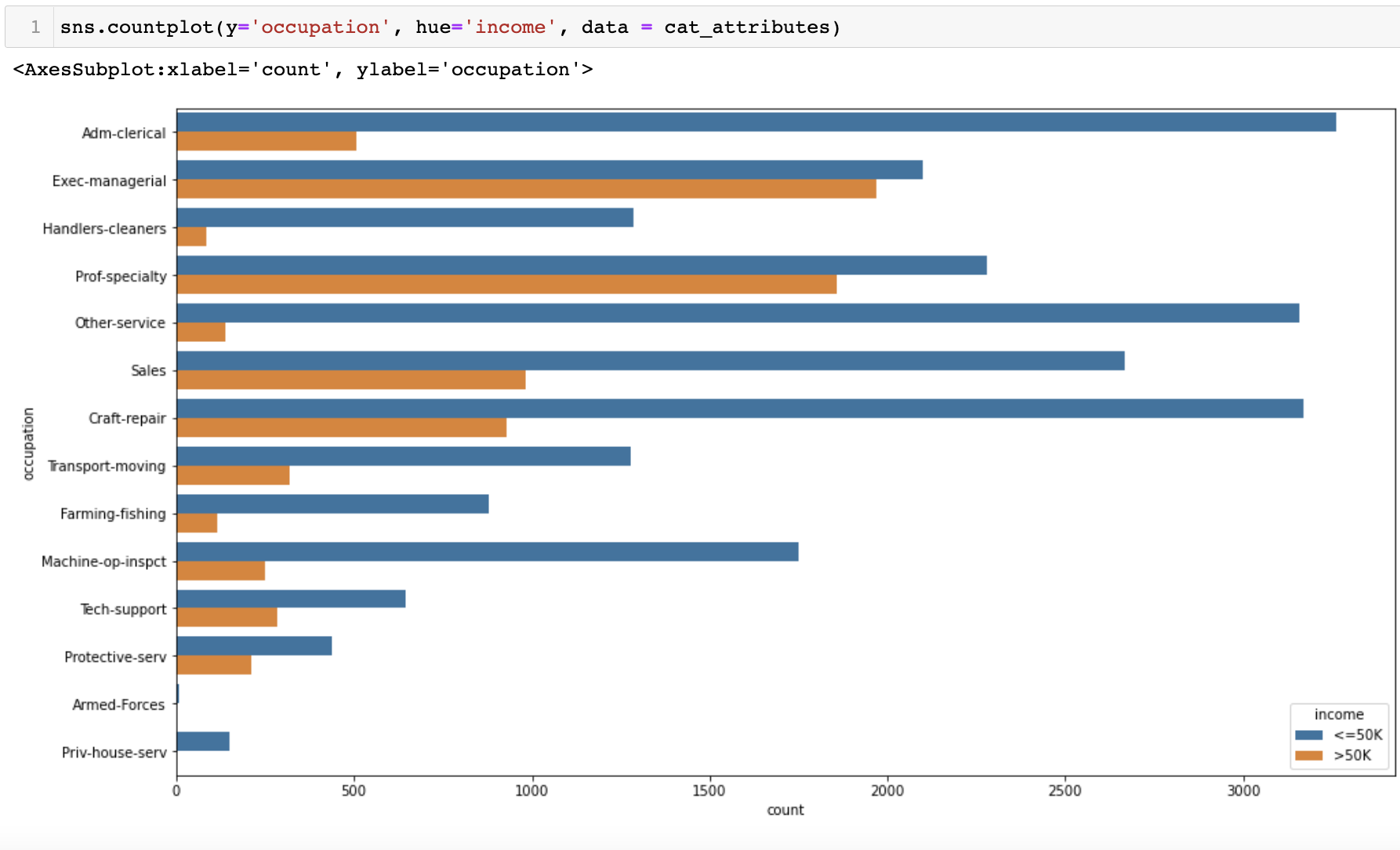
* Ни один из числовых атрибутов не имеет пропущенных значений
* Значения находятся в разных шкалах. Многие модели машинного обучения требуют, чтобы значения были в одном масштабе. Мы будем использовать StandardScaler из библиотеки sklearn для масштабирования функций.

1. Обработаем категориальные столбцы



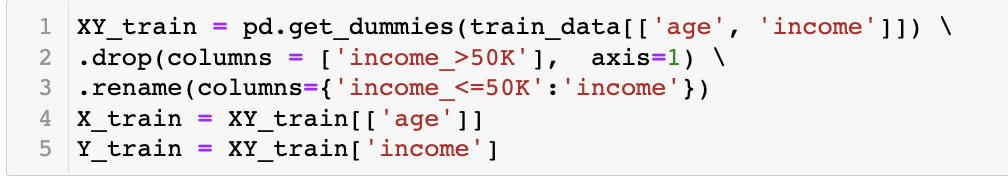
Провизуализиуем их, используя countplot из пакета seaborn.



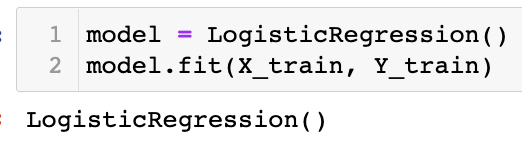


* Столбец «education» — это просто строковое представление столбца «education-num». Мы опустим колонку education .
* Переменные workClass , occupation , native-country имеют пропущенные значения. Мы заменим отсутствующие значения в каждом столбце наиболее часто встречающимся значением этого столбца.

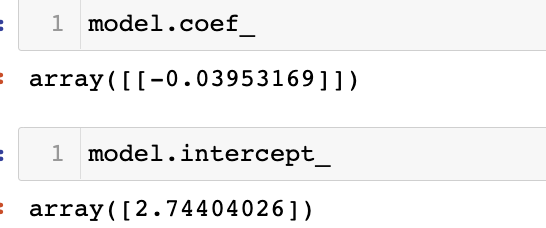
1. Выберем столбцы для построения логистической регрессии



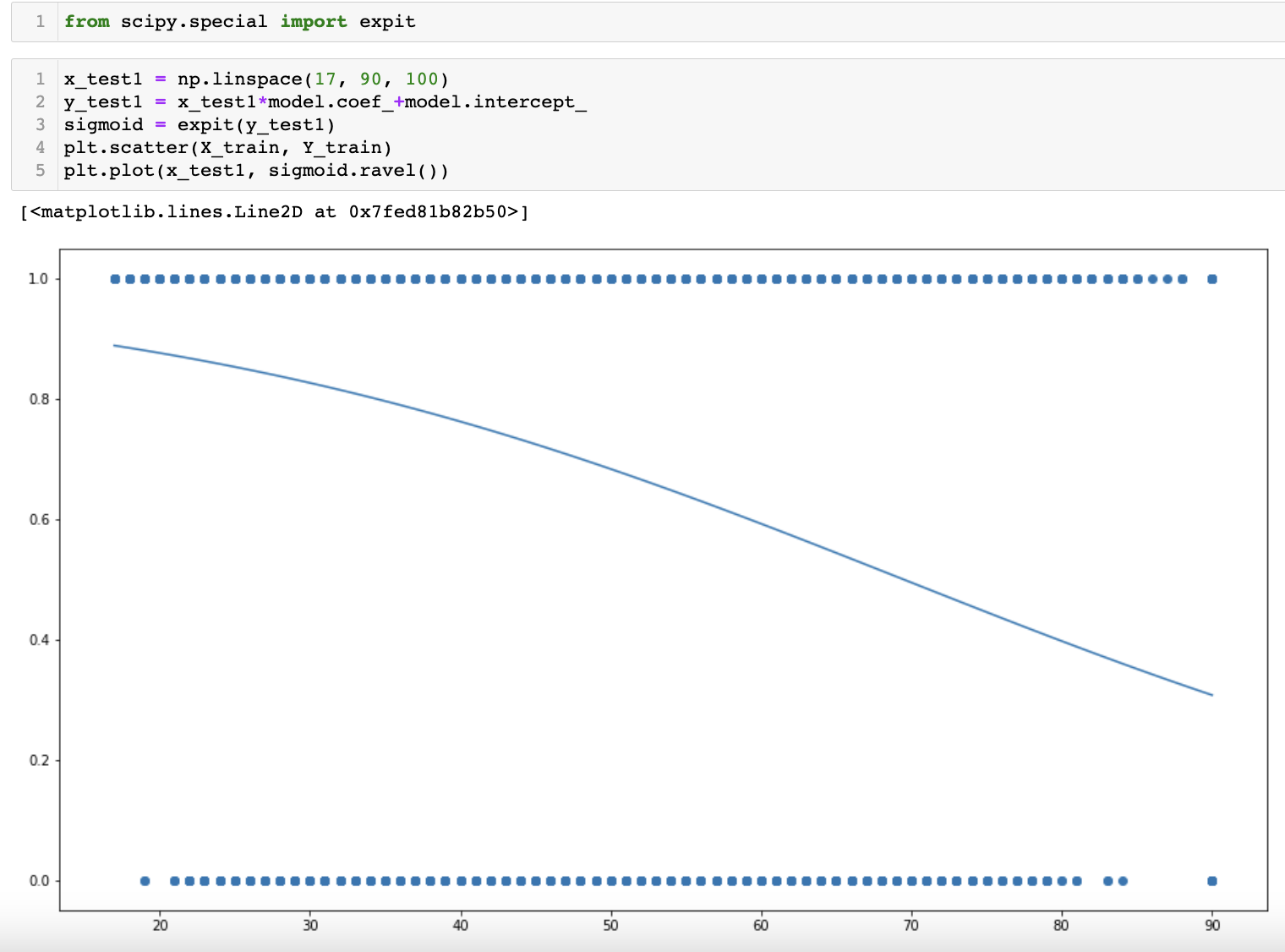
1. Создадим объект логистической регрессии и инициализируем процесс обучения



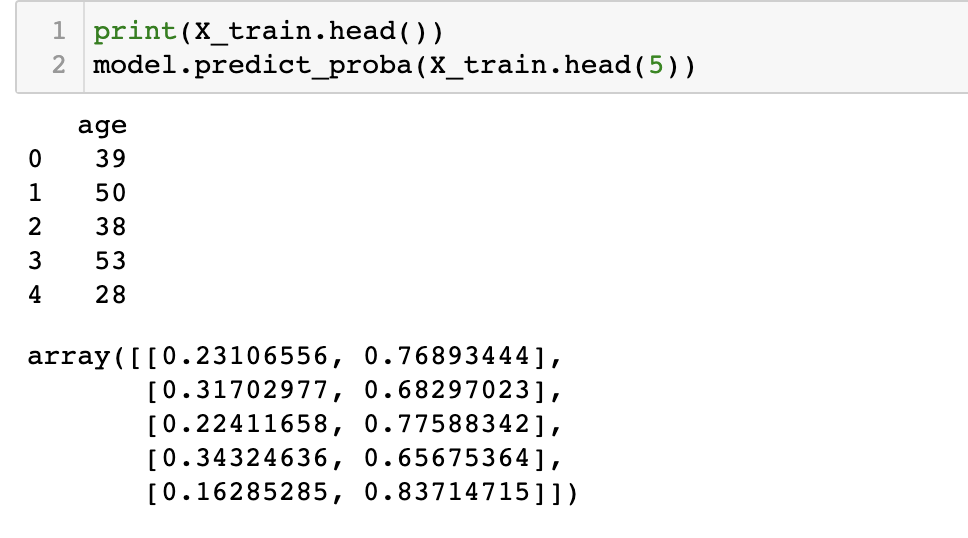
1. Найдем коэффициенты логистической регрессионной модели



1. Построим логистическую кривую и совместим её с облаком точек

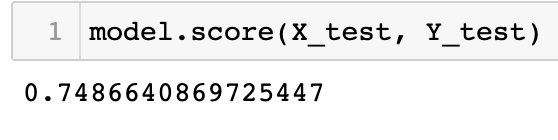


1. Выполним прогноз переменной отклика для нескольких объектов обучающей выборки



Для первых пяти объектов из обучающей выборки получили прогноз: все пять объектов с вероятностями от 16% до 34% работают

1. Вывод: точность нашей логистической регрессионной модели составила 74,86%



**Приложение**

|  |
| --- |
| import numpy as np  import pandas as pd  import io  import requests  import seaborn as sns  from matplotlib import pyplot as plt  import pickle  import os  from pandas.api.types import CategoricalDtype  from sklearn.base import BaseEstimator, TransformerMixin  from sklearn.pipeline import Pipeline  from sklearn.metrics import accuracy\_score  from sklearn.linear\_model import LogisticRegression  from sklearn.preprocessing import LabelEncoder  from sklearn.model\_selection import GridSearchCV  from sklearn.metrics import classification\_report  from sklearn.metrics import confusion\_matrix  from sklearn.preprocessing import StandardScaler  from sklearn.pipeline import FeatureUnion  from sklearn.model\_selection import cross\_val\_score  %matplotlib inline  plt.rcParams['figure.figsize'] = (16, 9) |
| columns = ["age", "workClass", "fnlwgt", "education", "education\_num",  "marital\_status", "occupation", "relationship",  "race", "sex", "capital\_gain", "capital\_loss", "hours\_per\_week",  "native\_country", "income"]  train\_data = pd.read\_csv('data/adult.data', names=columns,  sep=' \*, \*', na\_values='?', engine='python')  test\_data = pd.read\_csv('data/adult.test', names=columns,  sep=' \*, \*', skiprows=1, na\_values='?', engine='python') |
| train\_data.head() |
| train\_data.info() |
| num\_attributes = train\_data.select\_dtypes(include=['int'])  print(num\_attributes.columns) |
| num\_attributes.hist(figsize=(10,10)) |
| train\_data.describe() |
| cat\_attributes = train\_data.select\_dtypes(include=['object'])  print(cat\_attributes.columns) |
| sns.countplot(y='workClass', hue='income', data = cat\_attributes) |
| sns.countplot(y='occupation', hue='income', data = cat\_attributes) |
| train\_data.head() |
| XY\_train = pd.get\_dummies(train\_data[['age', 'income']]) \  .drop(columns = ['income\_>50K'], axis=1) \  .rename(columns={'income\_<=50K':'income'})  X\_train = XY\_train[['age']]  Y\_train = XY\_train['income'] |
| model = LogisticRegression()  model.fit(X\_train, Y\_train) |
| model.coef\_ |
| model.intercept\_ |
| from scipy.special import expit |
| x\_test1 = np.linspace(17, 90, 100)  y\_test1 = x\_test1\*model.coef\_+model.intercept\_  sigmoid = expit(y\_test1)  plt.scatter(X\_train, Y\_train)  plt.plot(x\_test1, sigmoid.ravel()) |
| print(X\_train.head())  model.predict\_proba(X\_train.head(5)) |
| model.predict([[50]]) |
| XY\_test = pd.get\_dummies(test\_data[['age', 'income']]) \  .drop(columns = ['income\_>50K.'], axis=1) \  .rename(columns={'income\_<=50K.':'income'})  X\_test = XY\_test[['age']]  Y\_test = XY\_test['income'] |
| model.score(X\_test, Y\_test) |