|  |
| --- |
|  |

**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский ядерный университет «МИФИ»

**Обнинский институт атомной энергетики –**

филиал федерального государственного автономного образовательного учреждения высшего

образования "Национальный исследовательский ядерный университет «МИФИ»

(ИАТЭ НИЯУ МИФИ)

Отделение Интеллектуальные кибернетические системы

**Лабораторная работа № 3**

**Средства визуализации данных в языках программирования**

Выполнила:

студентка гр. ИС-М18 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Яковлева Е.А.

Принял:

профессор ОИКС \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сальников Н.Л.

Обнинск, 2018

Цель работы: выбрать интересующий язык программирования и изучить средства визуализации данных. Выбрано: научные библиотеки Plot.ly графиков для Python, R, MATLAB, Perl, Julia, Arduino и REST.

Выполнение работы:

Плотли, также известный по его URL-адресу, Plot.ly (https://plot.ly/) - техническая компьютерная компания, которая разрабатывает онлайн- аналитику данных и инструменты визуализации. Plotly предоставляет онлайн-графические, аналитические и статистические инструменты для отдельных лиц и компаний, а также научные библиотеки графиков для Python, R, MATLAB, Perl, Julia, Arduino и REST.

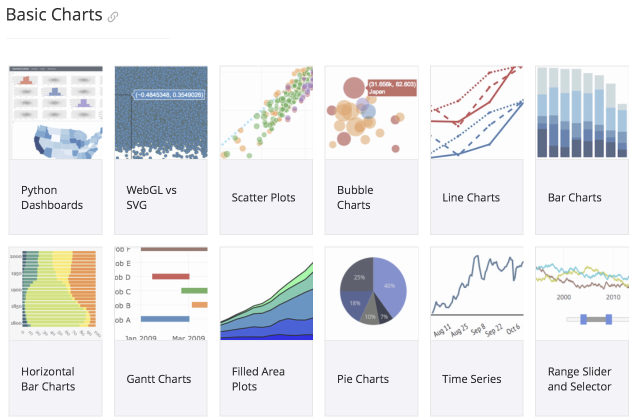


Рисунок 1 – Виды графиков, предоставляемых библиотекой Plot.ly

Рассмотрим построение графиков с помощью библиотек плотли на примере языков JavaScript, Python, Dash, R, React.

1. Первый график – столбчатая диаграмма. Для демонстрации была выбрана диаграмма с разделениями столбцов по цветам.

Исходный код (JavaScript) для столбчатой диаграммы:

var data = [{

x: ['giraffes', 'orangutans', 'monkeys'],

y: [20, 14, 23],

name: 'SF Zoo’,

type: 'bar',

marker: {color: '#19d3f3'}

}, {

x: ['giraffes', 'orangutans', 'monkeys'],

y: [12, 18, 29],

name: 'LA Zoo',

type: 'bar',

marker: {color: '#ab63fa'}

}];

var layout = {

plot\_bgcolor: '#F5F7FA',

paper\_bgcolor: '#F5F7FA',

width: 500

};

Plotly.newPlot('myDiv', data, layout);

Результат выполнения кода представлен на рисунке 2.

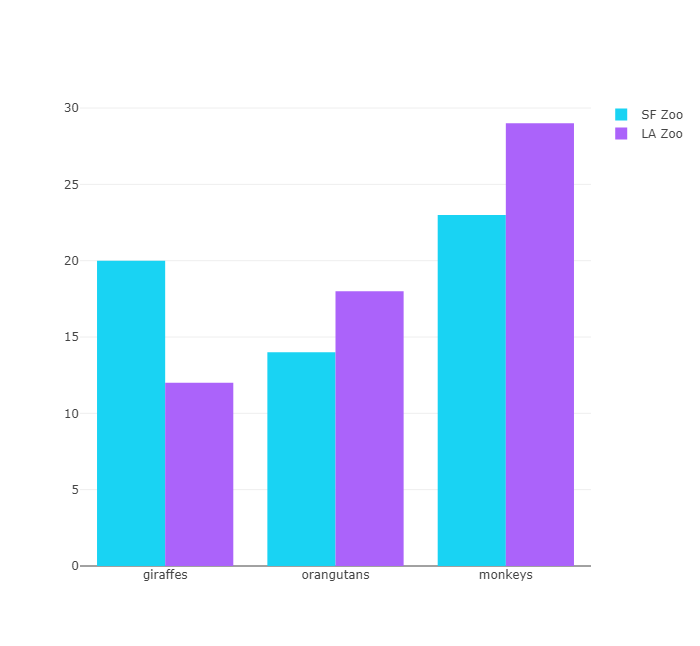


Рисунок 2 – Столбчатая диаграмма

1. Следующий график – точечная диаграмма.

Исходный код (Python) для точечной диаграммы.

import plotly.graph\_objs as go

import plotly.offline as py

import numpy as np

data = [go.Scatter(

y = np.random.randn(500),

mode='markers',

marker=dict(

size='16',

color = np.random.randn(500),

colorscale='Viridis',

showscale=True,

line={width: 0.5}

)

)]

layout = dict(

plot\_bgcolor ='#F5F7FA',

paper\_bgcolor = '#F5F7FA',

width = 500,

xaxis: {zeroline: false},

yaxis: {zeroline: false},

)

py.iplot( dict(data=data, layout=layout) )

Результат выполнения кода представлен на рисунке 3.

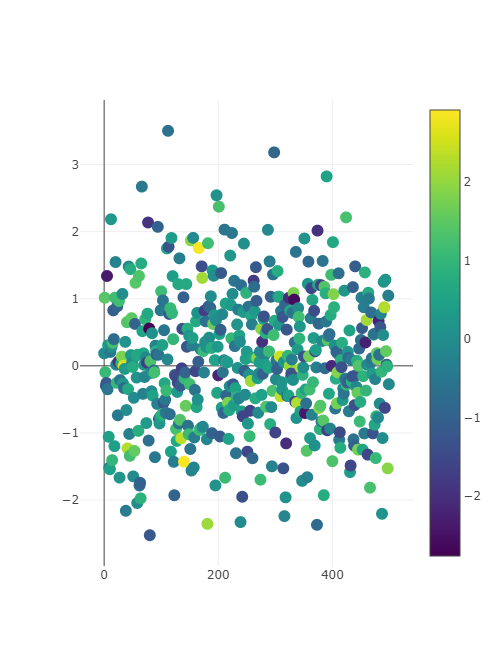


Рисунок 3 – Точечная диаграмма

1. Следующий график – box plot. Для демонстрации были сравнены оценки в зависимости от жанра фильма.

Исходный код (Python) для box plot.

N = float(len(top\_genres))

# cгенерируем цвета для визуализации

c = ['hsl('+str(h)+',50%'+',50%)' for h in np.linspace(0, 360, N)]

data = [{

'y': genres\_df[genres\_df.genre == top\_genres[i]].kp\_rating,

'type':'box',

'marker':{'color': c[i]},

'name': top\_genres[i]

} for i in range(len(top\_genres))]

layout = go.Layout(

title='Оценки фильмов',

yaxis = {'title': 'Оценка КиноПоиска'}

)

fig = go.Figure(data = data, layout = layout)

iplot(fig)

Результат выполнения кода представлен на рисунке 4.

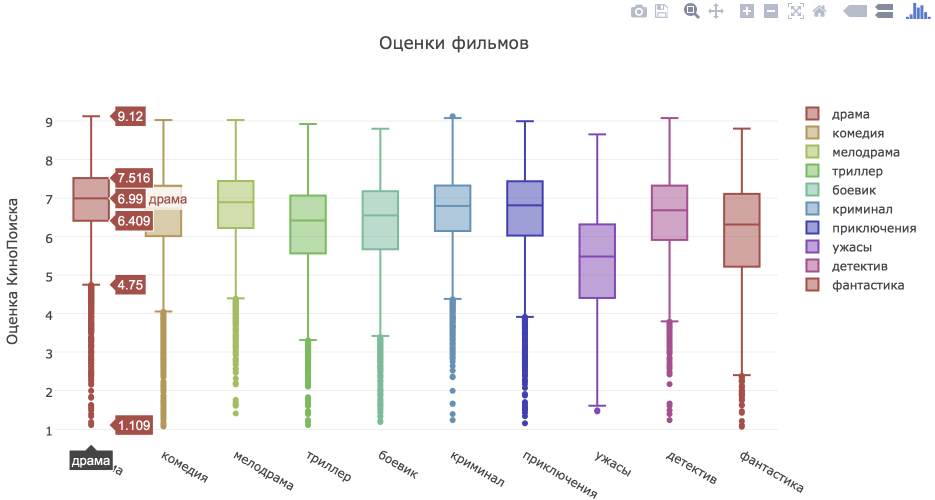


Рисунок 4 – Box plot

По графику видно, что больше всего выделяются низкими оценками фильмы-ужастики.

1. Следующий график – heatmap (тепловая карта). Для того чтобы посмотреть на то, какие жанры фильмов чаще встречаются вместе, построим heatmap.

Исходный код (Python) для heatmap.

genres\_coincidents = {}

for item in df.genres:

parsed\_genres = parse\_list(item)

for genre1 in parsed\_genres:

if genre1 not in genres\_coincidents:

genres\_coincidents[genre1] = defaultdict(int)

for genre2 in parsed\_genres:

genres\_coincidents[genre1][genre2] += 1

genres\_coincidents\_df = pd.DataFrame.from\_dict(genres\_coincidents).fillna(0)

# отнормируем таблицу на количество фильмов каждого жанра

genres\_coincidents\_df\_norm = genres\_coincidents\_df\

.apply(lambda x: x/genres\_df.groupby('genre').movie\_id.count(), axis = 1)

heatmap = go.Heatmap(

z = genres\_coincidents\_df\_norm.values,

x = genres\_coincidents\_df\_norm.index.values,

y = genres\_coincidents\_df\_norm.columns

)

layout = go.Layout(

title = 'Связанные жанры'

)

fig = go.Figure(data = [heatmap], layout = layout)

iplot(fig)

Результат выполнения кода представлен на рисунке 5.

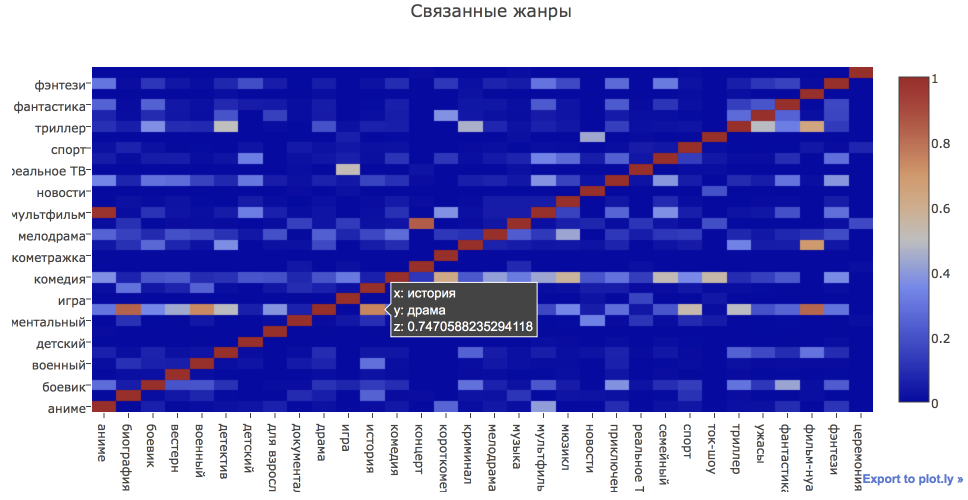


Рисунок 5 – Heatmap

Как видно из графика - 74,7% исторических фильмов также имеют тег драма.

1. Следующий график – линейная диаграмма.

Исходный код (Dash) для линейной диаграммы.

import dash

from dash.dependencies import Input, Output

import dash\_core\_components as dcc

import dash\_html\_components as html

from pandas\_datareader import data as web

from datetime import datetime as dt

app = dash.Dash()

app.layout = html.Div([

dcc.Dropdown(

id='my-dropdown',

options=[

{'label': 'Coke', 'value': 'COKE'},

{'label': 'Tesla', 'value': 'TSLA'},

{'label': 'Apple', 'value': 'AAPL'}

],

value='COKE'

),

dcc.Graph(id='my-graph')

])

@app.callback(Output('my-graph', 'figure'), [Input('my-dropdown', 'value')])

def update\_graph(selected\_dropdown\_value):

df = web.DataReader(

selected\_dropdown\_value, data\_source='google',

start=dt(2017, 1, 1), end=dt.now())

return {

'data': [{

'x': df.index,

'y': df.Close

}]

}

if \_\_name\_\_ == '\_\_main\_\_':

app.run\_server()

Результат выполнения кода представлен на рисунке 6.



Рисунок 6 – Линейная диаграмма

1. Исходный код (R) для стобчатой диаграммы.

library(plotly)

gg <- ggplot(diamonds, aes(carat)) +

geom\_histogram()

p <- ggplotly(gg)

p

Результат выполнения кода представлен на рисунке 7.

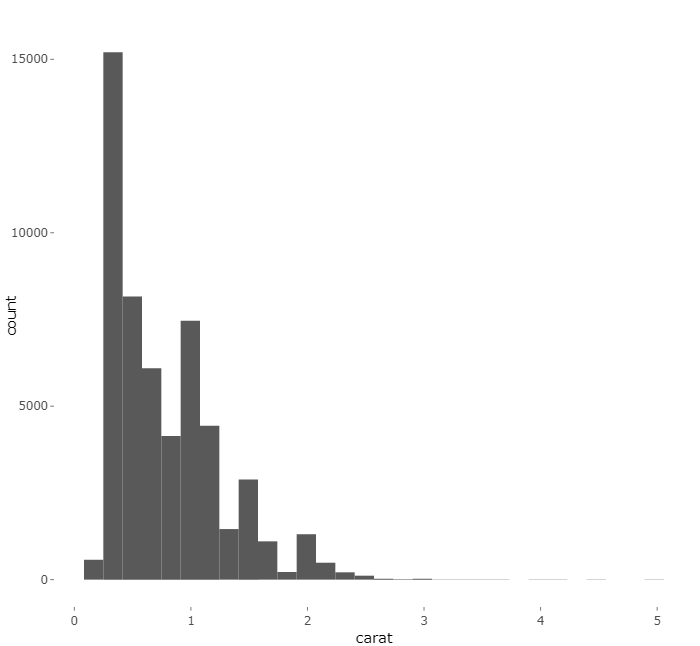


Рисунок 7 – Столбчатая диаграмма

1. Исходный код (React) для линейной диаграммы.

Результат выполнения кода представлен на рисунке 8.

import Plot from 'react-plotly.js'

const trace1 = {

type: 'line',

x: [1, 2, 3, 4 ],

y: [1, 3, 2, 5 ],

};

export default () => (

<Plot

data={[trace1]}

/>)

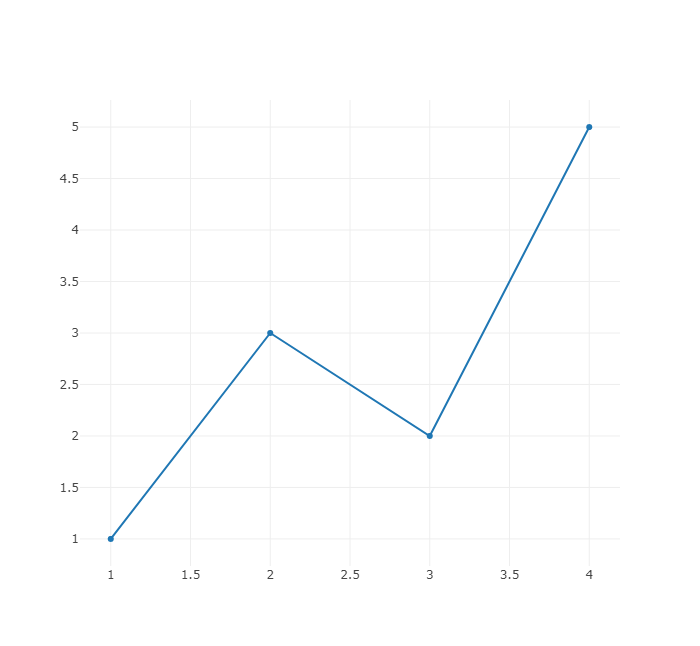


Рисунок 8 – Линейная диаграмма

Вывод: в результате выполнения работы были изученв научные библиотеки Plot.ly графиков для Python, R, MATLAB, Perl, Julia, Arduino и REST.Были продемонстрированы примеры построения графиков таких типов как: линейный, гистограмма, box plot, столбчатая и тепловая карта.

Список использованных источников

1. Plot.ly [Электронный ресурс]. URL: https://plot.ly/ [Доступ от 12.12.2018]