

Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



## **Лабораторная работа № 1**

**по дисциплине «Методы машинного обучения»**

**Создание «истории о данных»**

ИСПОЛНИТЕЛЬ:

студент ИУ5-23М

Бондаренко И. Г.

ПРЕПОДАВАТЕЛЬ:

Гапанюк Ю. Е.

\_\_\_ " \_\_\_\_\_ " 2024 г.

Москва, 2024

---

## ✓ Задание лабораторной работы

- Выбрать набор данных (датасет).
- Создать "историю о данных" в виде юпитер-ноутбука, с учетом следующих требований:
  - История должна содержать не менее 5 шагов (где 5 - рекомендуемое количество шагов). Каждый шаг содержит график и его текстовую интерпретацию.
  - На каждом шаге наряду с удачным итоговым графиком рекомендуется в юпитер-ноутбуке оставлять результаты предварительных "неудачных" графиков.
  - Не рекомендуется повторять виды графиков, желательно создать 5 графиков различных видов.
  - Выбор графиков должен быть обоснован использованием методологии data-to-viz. Рекомендуется учитывать типичные ошибки построения выбранного вида графика по методологии data-to-viz. Если методология Вами отвергается, то просьба обосновать Ваше решение по выбору графика.
  - История должна содержать итоговые выводы. В реальных "историях о данных" именно эти выводы представляют собой основную ценность для предприятия.
- Сформировать отчет и разместить его в своем репозитории на github.

---

[+ Код](#)[+ Текст](#)


## ✓ Выполнение работы

### ✓ Импорт библиотек

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import scipy.stats as stats
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.neighbors import KNeighborsRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import train_test_split
from IPython.display import Image
%matplotlib inline
sns.set(style="ticks")
```

## ✓ Подключение Google Диска для работы с Google Colab

```
from google.colab import drive
drive.mount('/content/drive')
```

 Mounted at /content/drive

### ✓ Чтение данных

```
data = pd.read_csv('/content/drive/MyDrive/MMO/PopularSpotifySongs.csv', encoding='unicode_escape')
```

```
data.head()
```

	track_name	artist(s)_name	artist_count	released_year	released_month	released_
0	Seven (feat. Latto) (Explicit Ver.)	Latto, Jung Kook	2	2023	7	
1	LALA	Myke Towers	1	2023	3	
2	vampire	Olivia Rodrigo	1	2023	6	
3	Cruel Summer	Taylor Swift	1	2019	8	
4	WHERE SHE GOES	Bad Bunny	1	2023	5	

5 rows × 24 columns

data.shape

(953, 24)

Набор содержит как категориальные признаки, так и числовые.

## История о данных

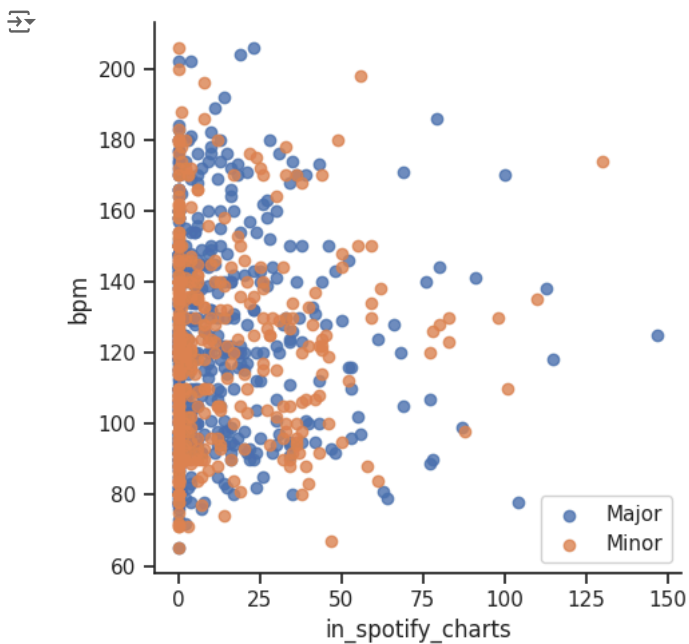
Возьмем признаки: mode (категориальный), bpm (числовой) и in\_spotify\_charts (числовой). По методологии data\_to\_viz построим Scatter Plot (Точечный график), 2D Density (Двумерное распределение), Box Plot (Ящик с усами), Violin Plot и Correlogram.

```
x = data["in_spotify_charts"]
y = data["bpm"]
z = data["mode"]
d = data[["in_spotify_charts", "bpm", "mode"]]
```

```
# Use the 'hue' argument to provide a factor variable
sns.lmplot(x="in_spotify_charts", y="bpm", data=d, fit_reg=False, hue="mode", legend=False)
```

```
# Move the legend to an empty part of the plot
plt.legend(loc='lower right')
```

```
plt.show()
```



Точечный график (Scatter Plot) показывает зависимость между двумя числовыми признаками - horsepower и price. По графику можно сделать вывод о том, что в среднем чем выше мощность автомобиля, тем выше и его стоимость. Цветными метками отображены распределение по carbody.

```

from scipy.stats import kde

a = x, y

# Create a figure with 6 plot areas
fig, axes = plt.subplots(ncols=5, nrows=1, figsize=(21, 5))

# Thus we can cut the plotting window in several hexbins
nbins = 20
axes[0].set_title('Hexbin')
axes[0].hexbin(x, y, gridsize=nbins, cmap=plt.cm.BuGn_r)

# 2D Histogram
axes[1].set_title('2D Histogram')
axes[1].hist2d(x, y, bins=nbins, cmap=plt.cm.BuGn_r)

# Evaluate a gaussian kde on a regular grid of nbins x nbins over data extents
k = kde.gaussian_kde(a)
xi, yi = np.mgrid[x.min():x.max():nbins*1j, y.min():y.max():nbins*1j]
zi = k(np.vstack([xi.flatten(), yi.flatten()]))

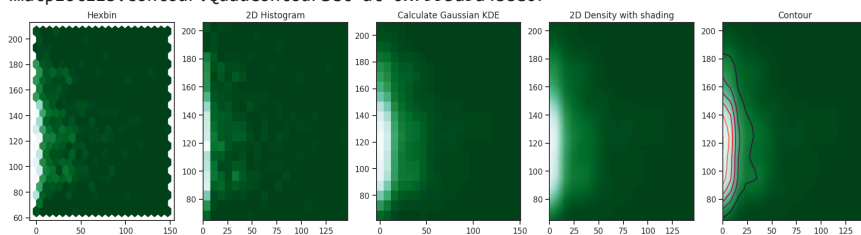
# plot a density
axes[2].set_title('Calculate Gaussian KDE')
axes[2].pcolormesh(xi, yi, zi.reshape(xi.shape), shading='auto', cmap=plt.cm.BuGn_r)

# add shading
axes[3].set_title('2D Density with shading')
axes[3].pcolormesh(xi, yi, zi.reshape(xi.shape), shading='gouraud', cmap=plt.cm.BuGn_r)

# contour
axes[4].set_title('Contour')
axes[4].pcolormesh(xi, yi, zi.reshape(xi.shape), shading='gouraud', cmap=plt.cm.BuGn_r)
axes[4].contour(xi, yi, zi.reshape(xi.shape) )

↳ <ipython-input-21-f8eb10e25fc7>:18: DeprecationWarning: Please use `gaussian_kde` frc
k = kde.gaussian_kde(a)
<matplotlib.contour.QuadContourSet at 0x7993a9a4b8e0>

```



Двумерное распределение по признакам price и horsepower показывает в цветном эквиваленте где больше всего есть значений данных. Чем ярче область, тем больше значений. По графикам видно, что наибольшее сосредоточенность данных присутствует в ценовом диапазоне до 20 000 и мощности до 120 л.с.

```
sns.boxplot( x=z, y=x )
```

```
<Axes: xlabel='mode', ylabel='in_spotify_charts'>
```

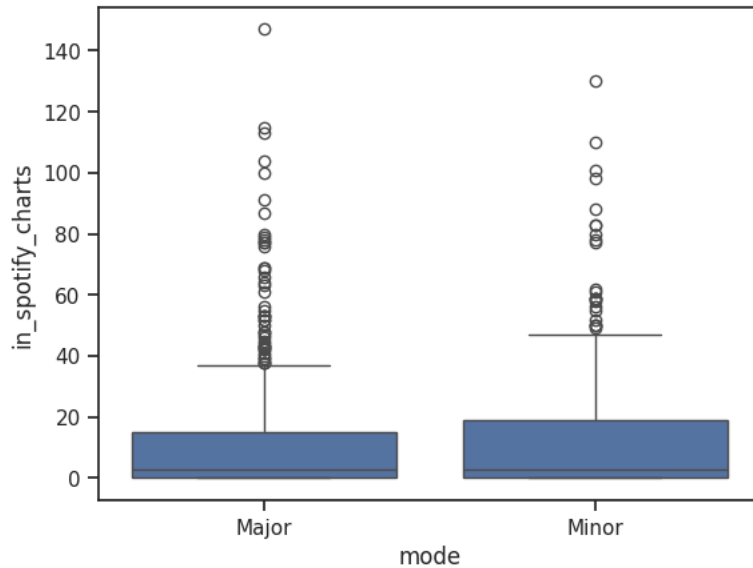
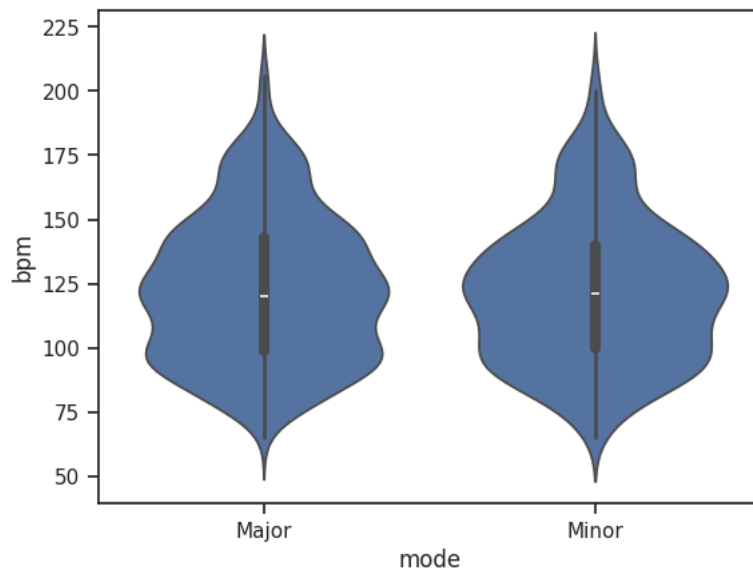


График "Ящик с усами" показывает распределение автомобилей по кузовам в рамках ценовых диапазонов. По графикам видно, что "хэтбеки" в основной массе недорогие по отношению к другим типам кузовов. "Купе" и "кабриолеты" - одни из самых дорогих типов автомобилей, причём "купе" располагаются в большом ценовом диапазоне - от 10 000 до 35 000.

```
sns.violinplot(x=z, y=y)
```

```
<Axes: xlabel='mode', ylabel='bpm'>
```



```
sns.pairplot(d, kind="scatter", hue="mode", markers=["o", "s", "D"], palette="Set2")  
plt.show()
```