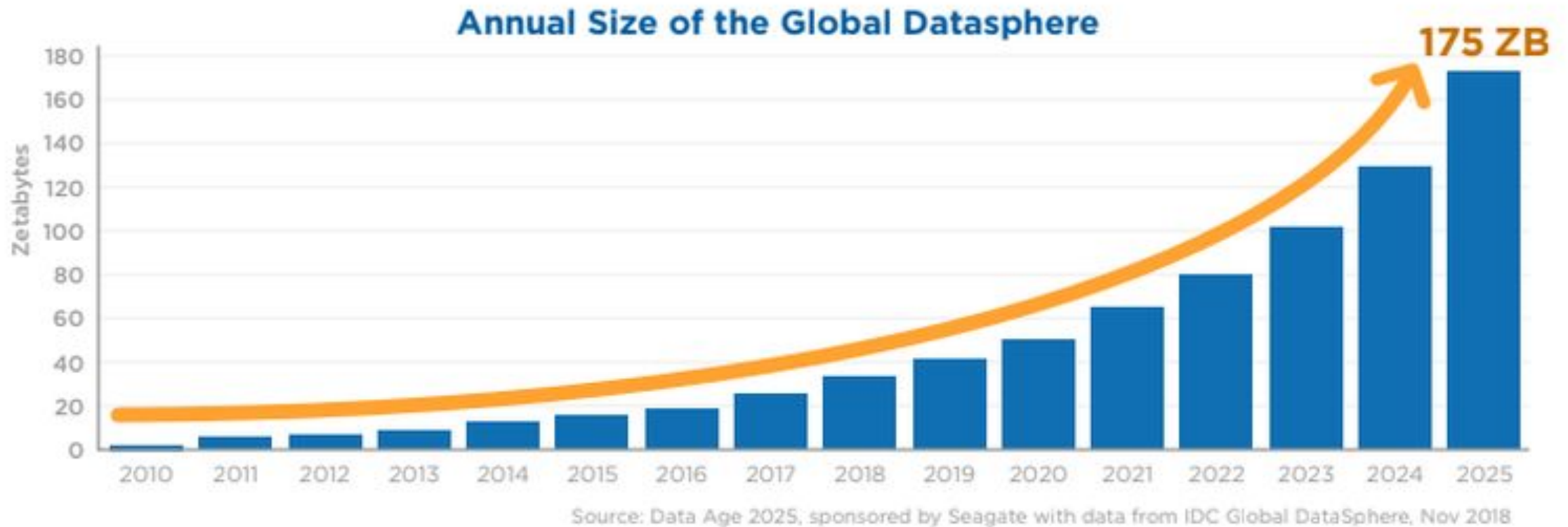


Основы Data science

Кластеризация

- **Data Science** — это работа с большими данными (*англ. Big Data*).

Figure 1 - Annual Size of the Global Datasphere



1 ZB = 10^{21} bytes

1 TB = 10^{12} bytes

Evolution of a Terabyte of Data 1956 - 2015

A decade by decade guide to storing 1TB of data from the 1950s to 2015

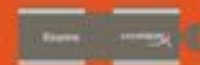


280MB

1960s

1TB

2013



Эволюция
в
области
хранения и
Обработки
данных

Кто такой Data Scientist?

DATA SCIENTIST MUST-HAVE SKILLS

MATH & STATISTICS

- Machine Learning
- Statistical Modeling
- Exploratory Analysis
- Clustering
- Regression Analysis

DOMAIN KNOWLEDGE & SOFT SKILLS

- Inclination towards business operations
- Keen on working with data
- Problem solver
- Strategic, proactive, and cooperative
- Interested in hacking



PROGRAMMING & DATABASE

- Computer Science Fundamentals
- Database Management System
- Data Visualization
- Python
- Big Data

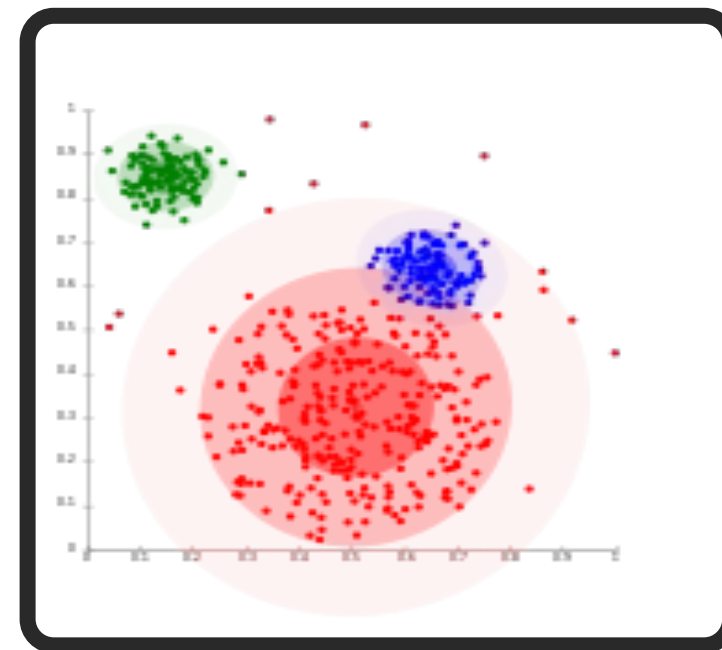
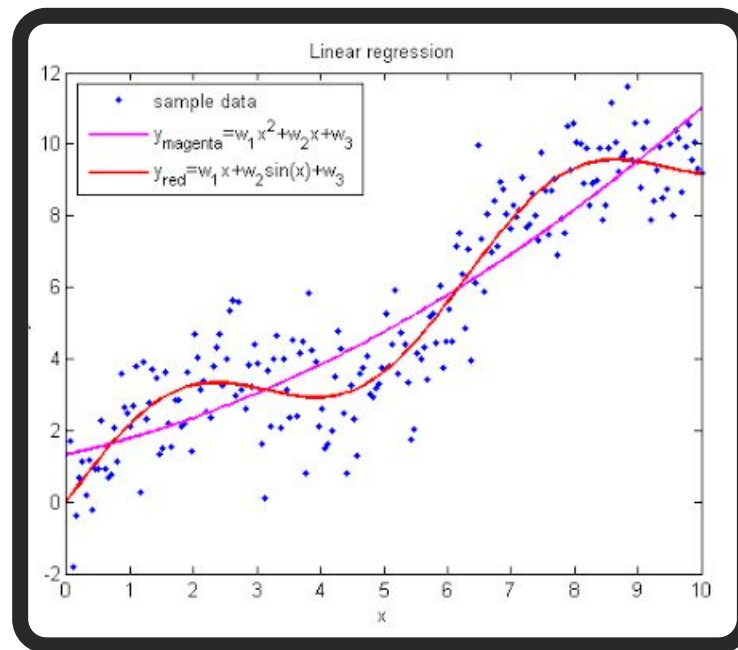
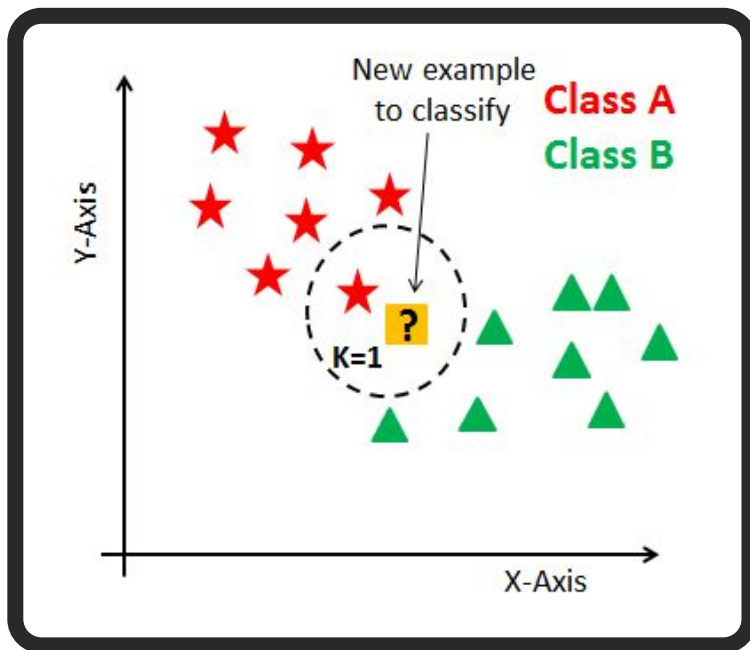
COMMUNICATION & VISUALIZATION

- Storytelling skills
- Convert data-based insights into decisions
- Collaborative with Sr. Management
- Knowledge of tools like Tableau
- Visual art design

Классификация

Регрессия

Кластеризация



Независимые переменные

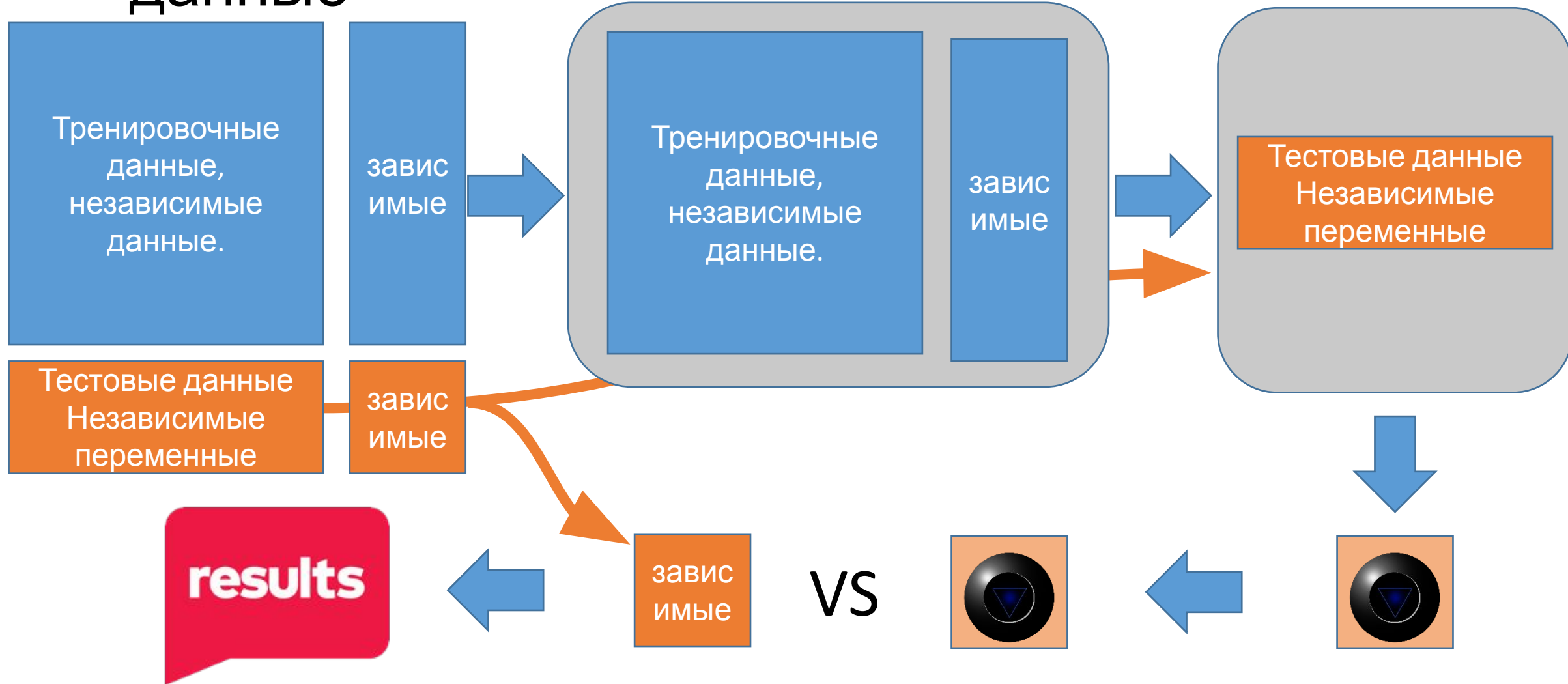
V - 1	V - 2	V - 3	V - 4	V - 5	V - 6	V - 7	V - 8	V - 9	V - 10
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value
value	value	value	value	value	value	value	value	value	value

Зависимая

Num	Class
1.003	cat
2.008	dog
7.256	dog
8.240	cat
3.001	cat
5.443	cat
2.754	dog
?	?

Исходные данные

Модель МО





Epoch
000,158

Learning rate
0.03

Activation
Tanh

Regularization
None

Regularization rate
0

Problem type
Classification

DATA

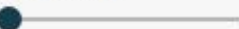
Which dataset do you want to use?



Ratio of training to test data: 50%



Noise: 0



Batch size: 10

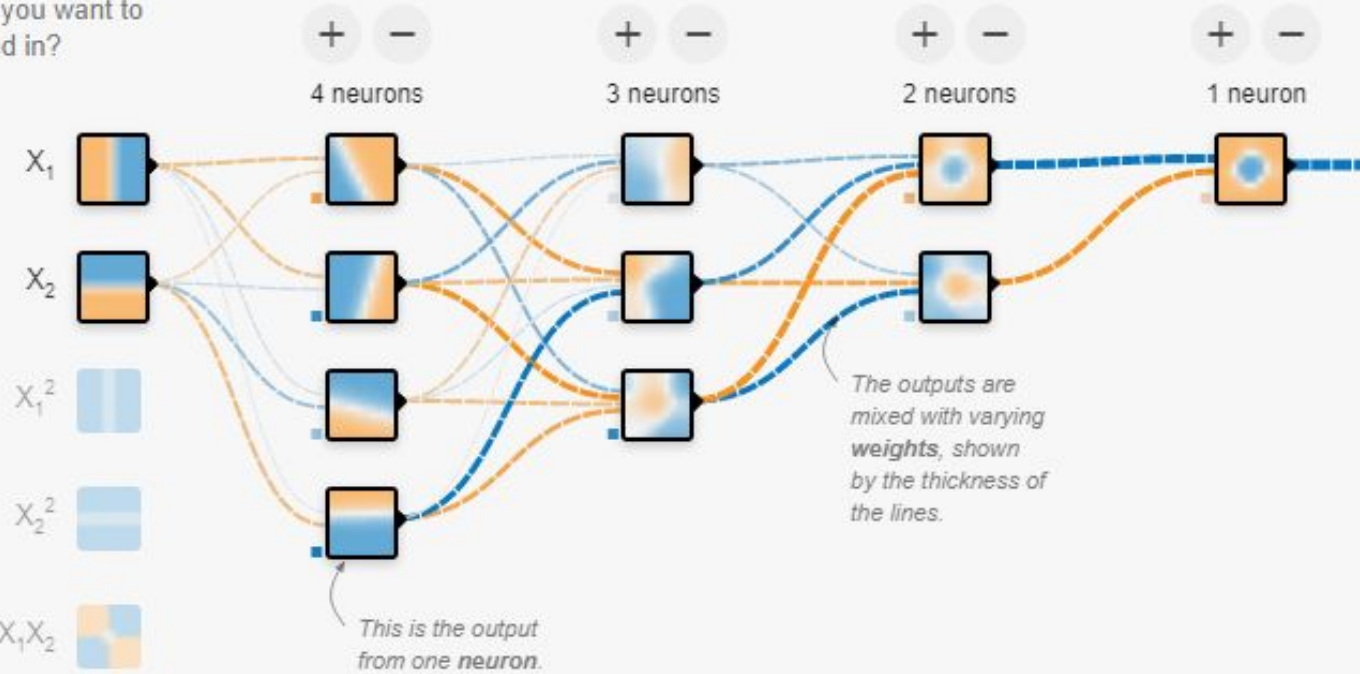


FEATURES

Which properties do you want to feed in?

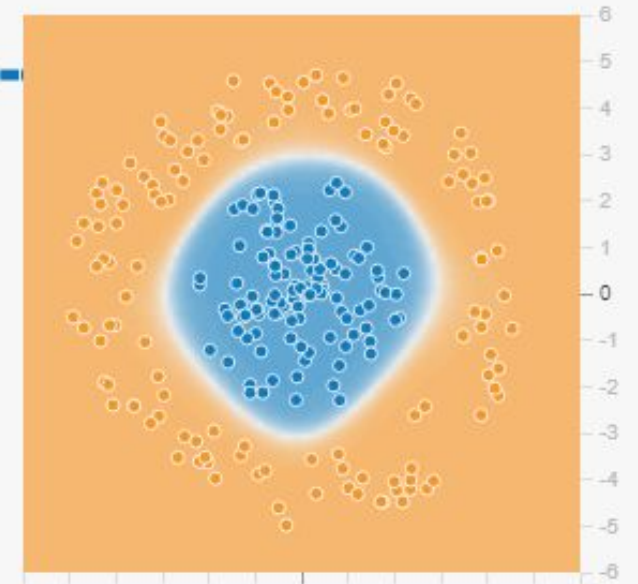
- X_1
- X_2
- X_1^2
- X_2^2
- X_1X_2

+ - 4 HIDDEN LAYERS



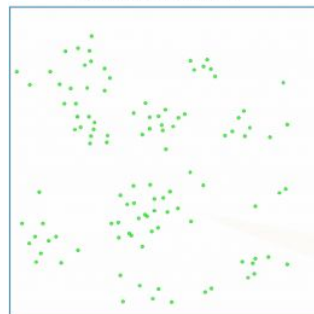
OUTPUT

Test loss 0.003
Training loss 0.001

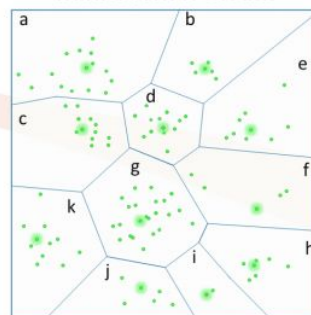


Кластеризация молекул

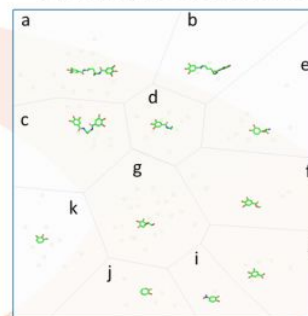
A. Molecules as fingerprints



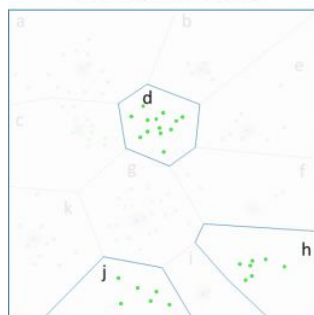
B. Clustering with Mini Batch K-means



C. Virtual Screening with 1 molecule/cluster



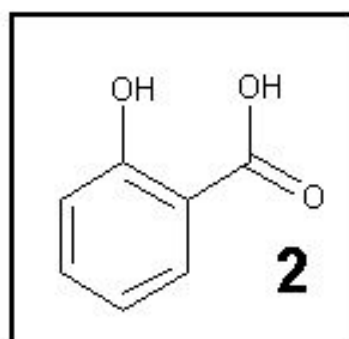
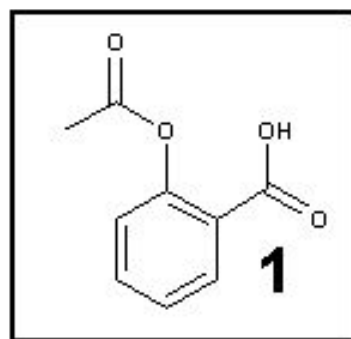
D. Identification of clusters for a second Virtual Screening



PharmScreen

	Cluster	Alignment	Score
Selected	h		0.87
	d		0.85
	j		0.81
	b		0.73
	k		0.69
	e		0.68
	f		0.62
	a		0.55
	i		0.50
	c		0.47
	g		0.46

Similarity Searching



1	1	1	0	1	1	0	1	0
2	1	1	0	1	0	0	0	0

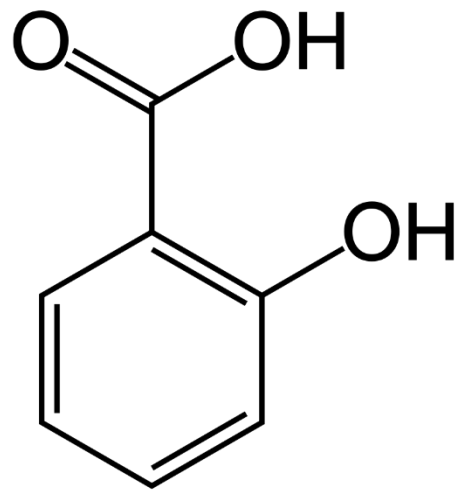
A = Number of bits set in both = 3

B = Number of bits set in (1), but not in (2) = 2

C = Number of bits set in (2), but not in (1) = 0

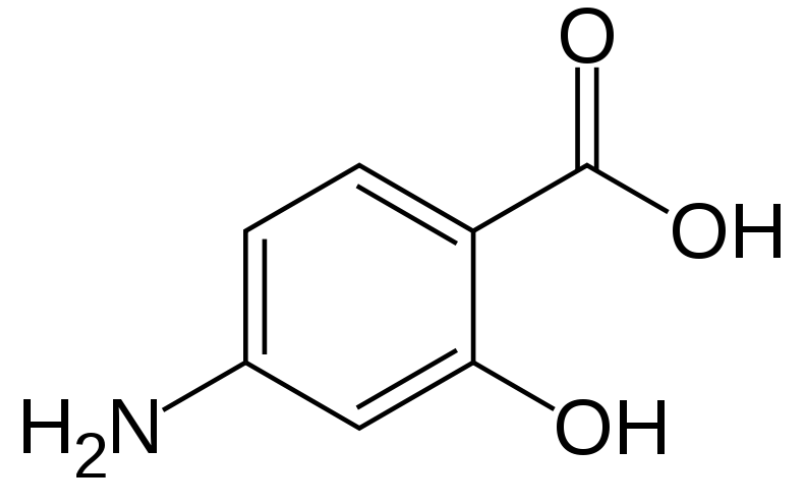
$$\text{TANIMOTO COEFFICIENT} = A / (A + B + C)$$
$$= 3 / (3 + 2 + 0) = 0.6 \text{ or } 60\%$$

Задача на Python



Salicylic acid

c1ccc(c(c1)C(=O)O)O



PASA

C1=CC(=C(C=C1N)O)C(=O)O

Tanimoto coefficient, Tanimoto distance

```
In [19]: from rdkit import Chem
         from rdkit.Chem import Descriptors
         from rdkit.Chem import rdFingerprintGenerator
```

```
In [20]: SA = 'c1ccc(c(c1)C(=O)O)O'
         PASA = 'C1=CC(=C(C=C1N)O)C(=O)O'
```

```
In [21]: SA = Chem.MolFromSmiles(SA)
         PASA = Chem.MolFromSmiles(PASA)
```

```
In [22]: rdkit_gen = rdFingerprintGenerator.GetRDKitFPGenerator(maxPath=5)
```

```
In [26]: SA_fp = rdkit_gen.GetFingerprint(SA)
         PASA_fp = rdkit_gen.GetFingerprint(PASA)
```

```
In [33]: sim = round(Chem.DataStructs.TanimotoSimilarity(SA_fp, PASA_fp), 3)
         sim
```

```
Out[33]: 0.811
```

```
In [36]: tanimoto_distance = round(1-sim, 3)
         tanimoto_distance
```

```
Out[36]: 0.189
```

```
data = pd.read_csv('Data_ML.csv')
```

```
compounds = []  
for _, chembl_id, smile in data[['molecule_chembl_id', 'Smiles']].itertuples():  
    compounds.append((Chem.MolFromSmiles(smile), chembl_id))  
compounds[:5]
```

```
[(<rdkit.Chem.rdchem.Mol at 0x280de08e7b0>, 'CHEMBL3640324'),  
 (<rdkit.Chem.rdchem.Mol at 0x280de08e530>, 'CHEMBL3640408'),  
 (<rdkit.Chem.rdchem.Mol at 0x280de08ee40>, 'CHEMBL3642557'),  
 (<rdkit.Chem.rdchem.Mol at 0x280dec69850>, 'CHEMBL3642442'),  
 (<rdkit.Chem.rdchem.Mol at 0x280dec693f0>, 'CHEMBL3955760')]
```

```
rdkit_gen = rdFingerprintGenerator.GetRDKitFPGenerator(maxPath=5)  
fingerprints = [rdkit_gen.GetFingerprint(mol) for mol, indx in compounds]
```

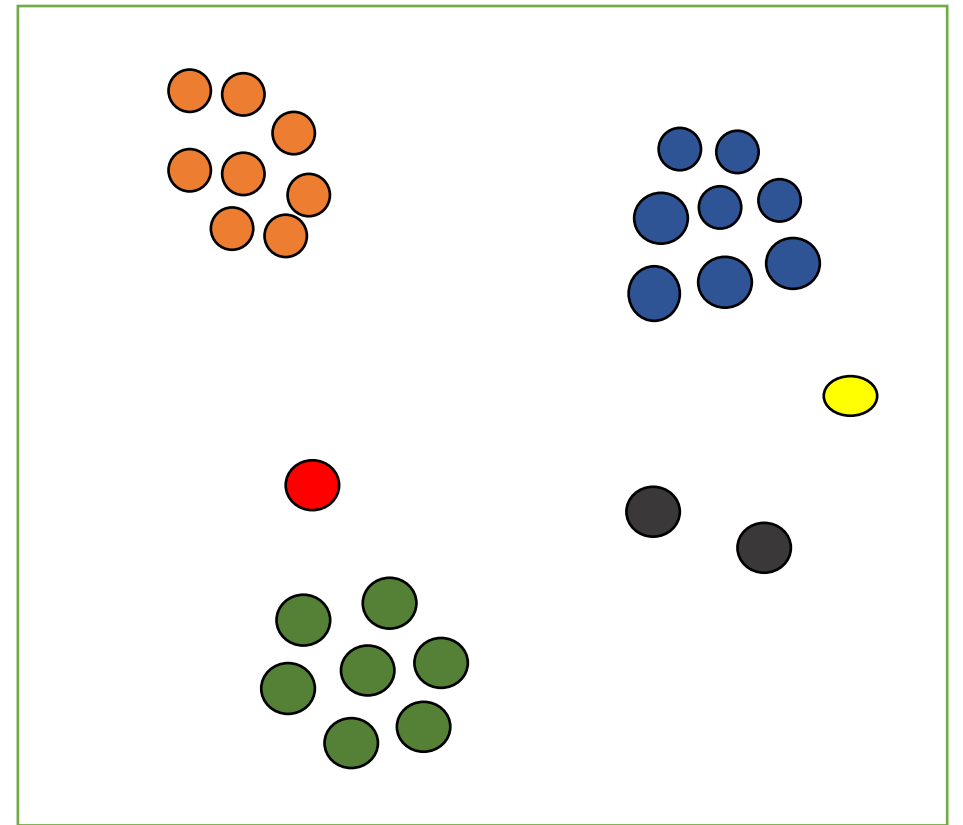
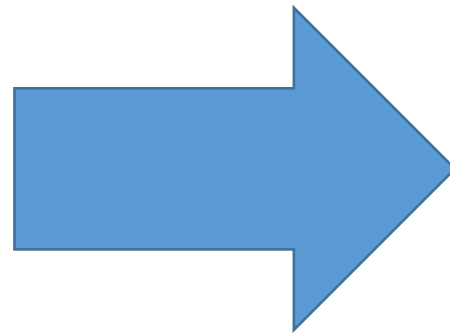
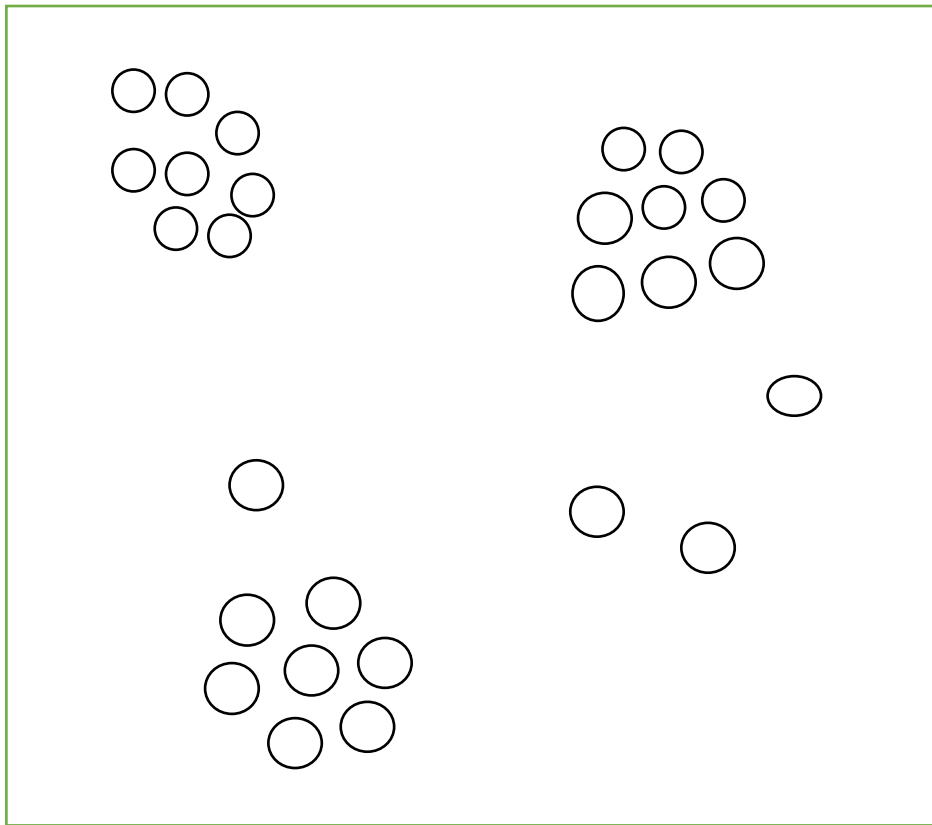
```
fingerprints[0:5]
```


Трудности...

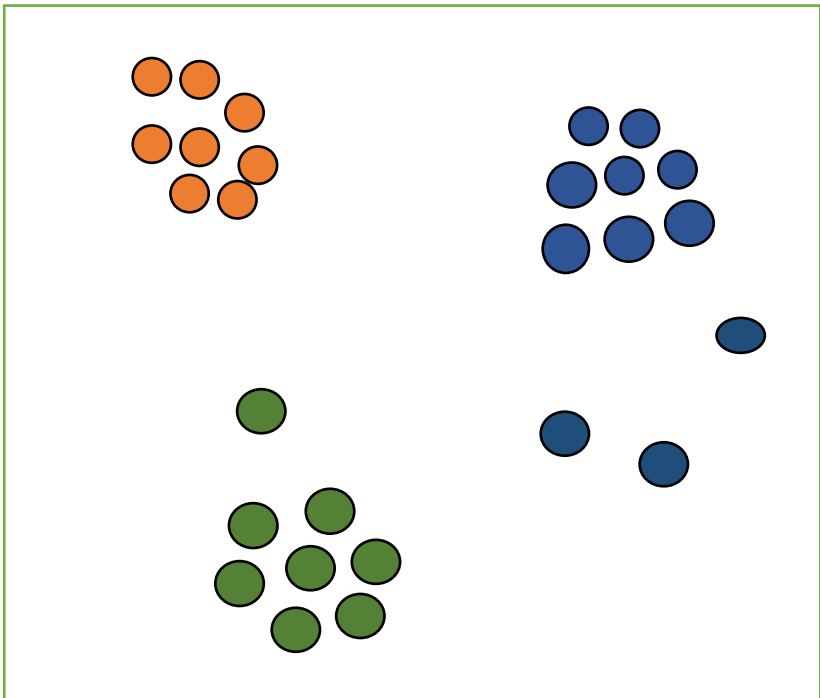
```
def tanimoto_distance_matrix(fp_list):  
    dissimilarity_matrix = []  
    for i in range(1, len(fp_list)):  
        similarities = DataStructs.BulkTanimotoSimilarity(fp_list[i], fp_list[:i])  
        dissimilarity_matrix.extend([1 - x for x in similarities])  
    return dissimilarity_matrix
```

```
tdm = tanimoto_distance_matrix(fingerprints)
```

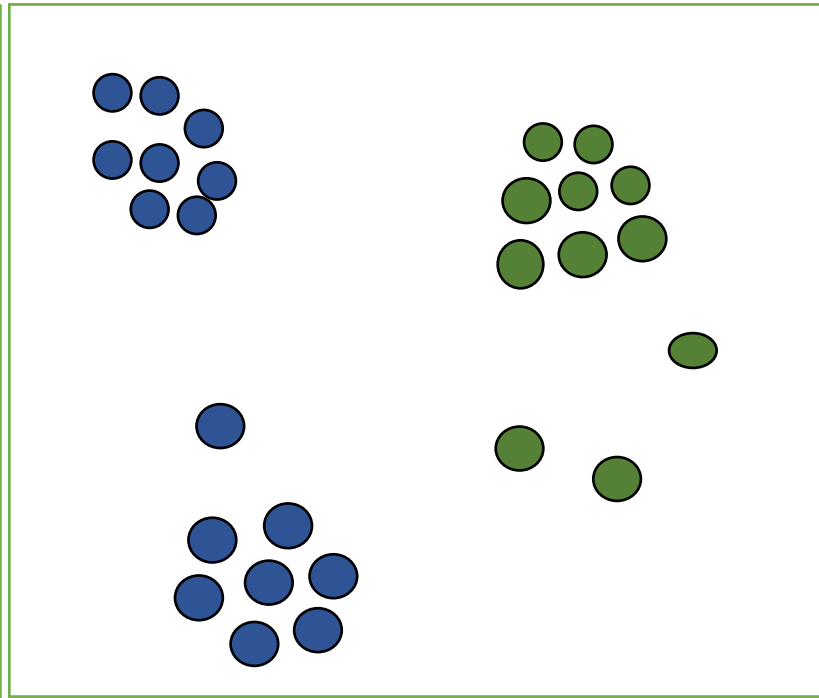
Принцип кластеризации



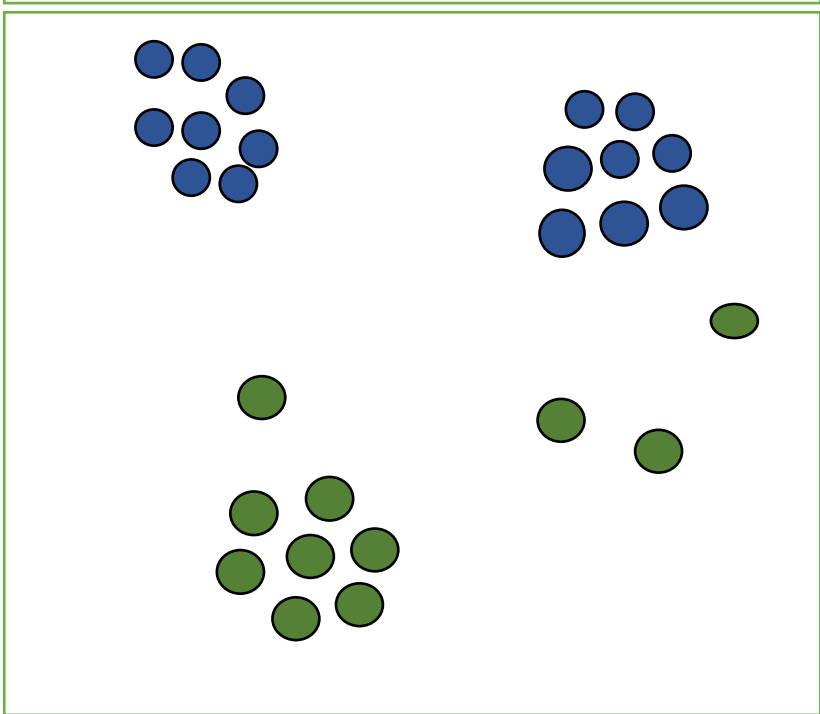
A



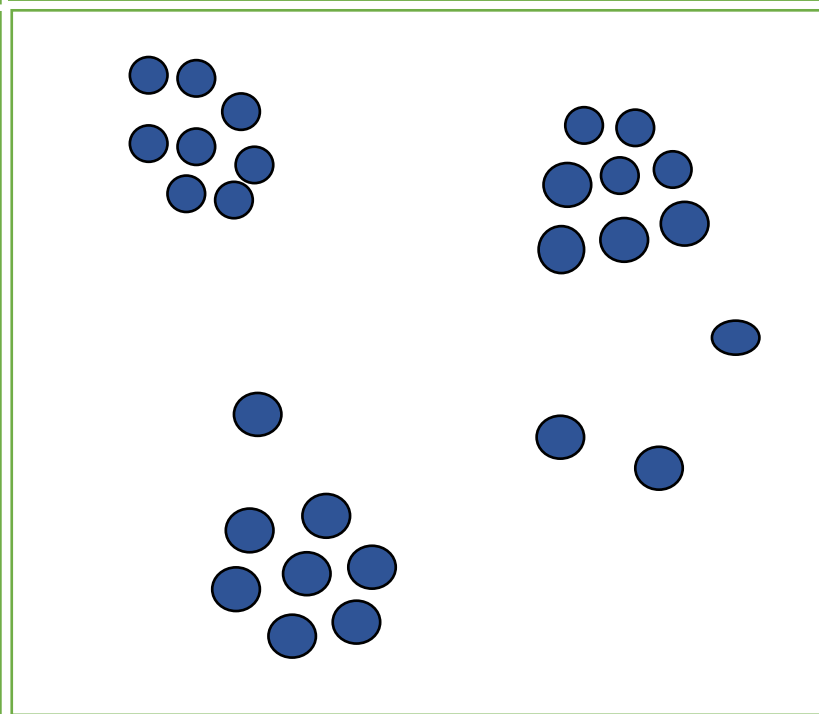
B



C



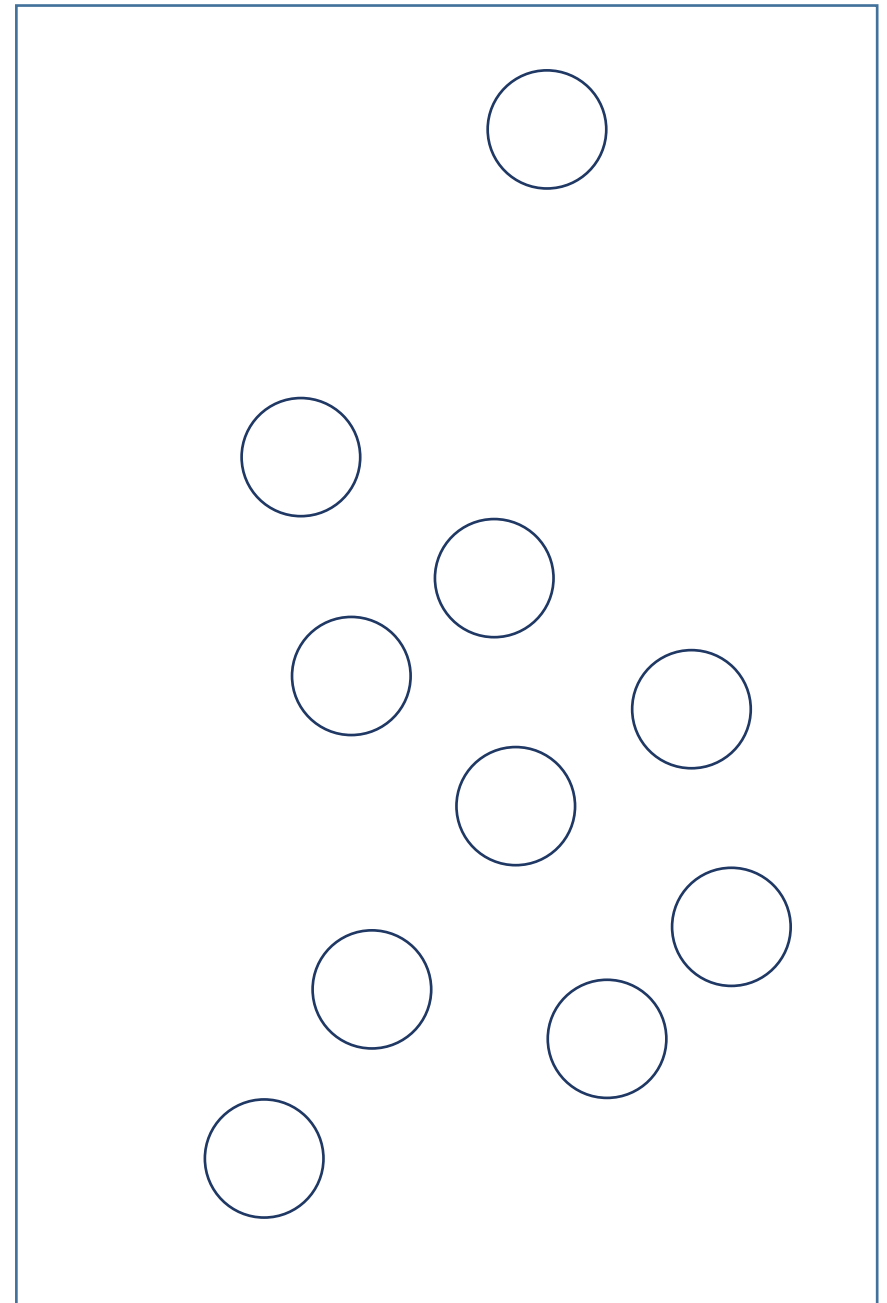
D

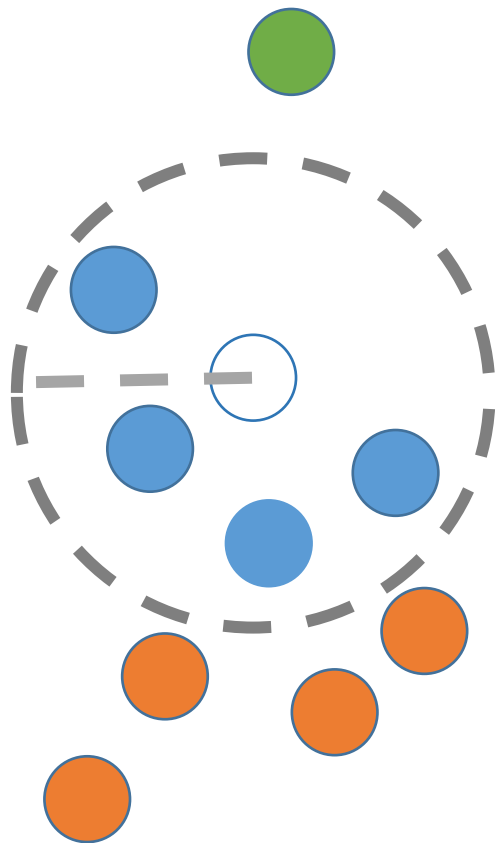


Для кластеризации
необходимы:

1. Расстояние
2. Центроиды

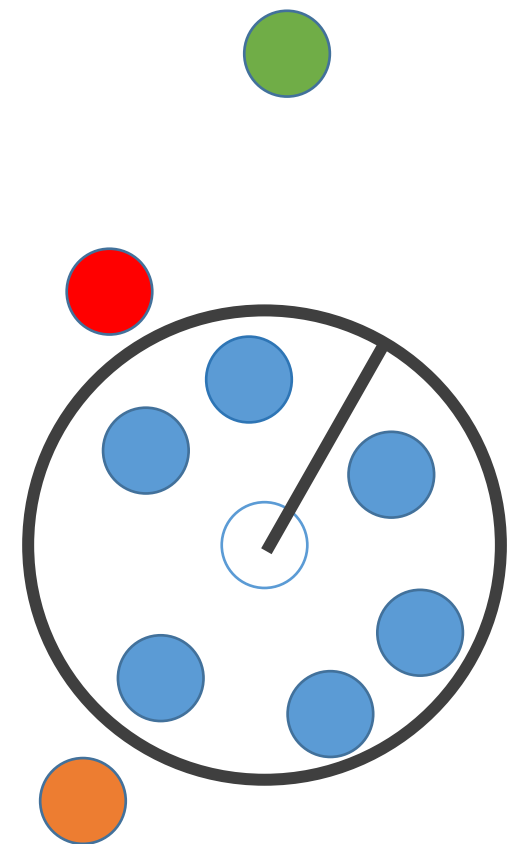
Цель – найти оптимальные
центроиды при данном
расстоянии



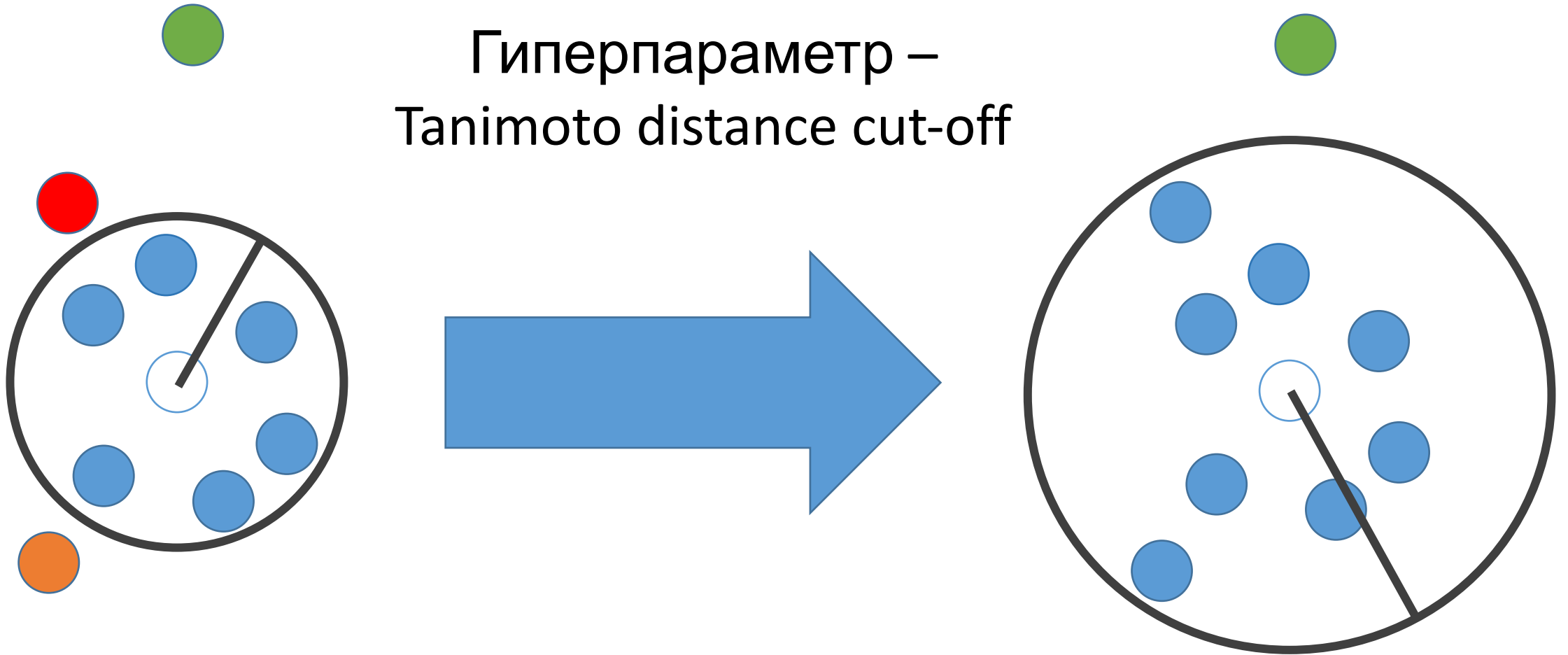


Не
оптимальный
центроид

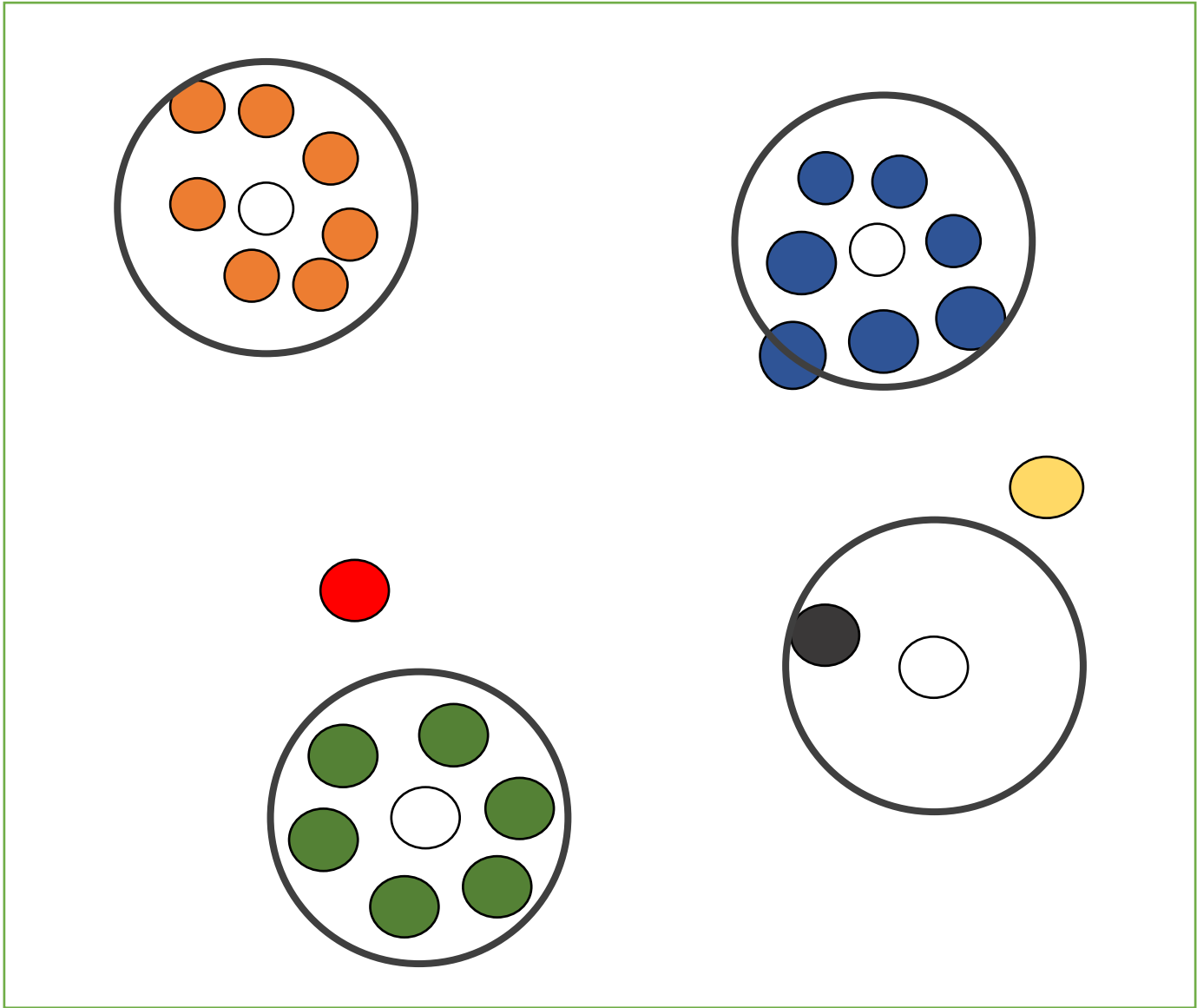
Оптимальны
й
центроид



Гиперпараметр – Tanimoto distance cut-off



Чем больше значение расстояния –
тем больше кластеры содержат элементов.
Тем меньше кластеров



```
def cluster_fingerprints(fingerprints, cutoff=0.2):
    tdm = tanimoto_distance_matrix(fingerprints)
    clusters = Butina.ClusterData(tdm, len(fingerprints), cutoff, isDistData=True)
    clusters = sorted(clusters, key=len, reverse=True)
    return clusters
```

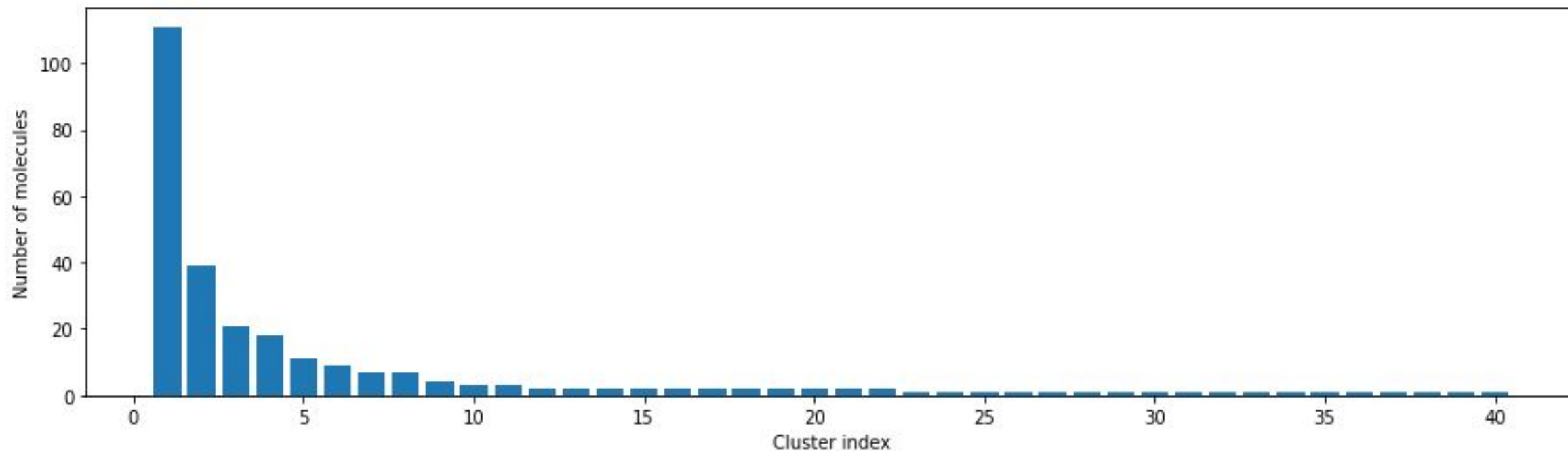
```
clusters = cluster_fingerprints(fingerprints, cutoff=0.35)
```

```
num_clust_g1 = sum(1 for c in clusters if len(c) == 1)
num_clust_g5 = sum(1 for c in clusters if len(c) > 5)
num_clust_g25 = sum(1 for c in clusters if len(c) > 25)
num_clust_g100 = sum(1 for c in clusters if len(c) > 100)
```

```
print("total # clusters: ", len(clusters))
print("# clusters with only 1 compound: ", num_clust_g1)
print("# clusters with >5 compounds: ", num_clust_g5)
print("# clusters with >25 compounds: ", num_clust_g25)
print("# clusters with >100 compounds: ", num_clust_g100)
```

```
total # clusters: 18
# clusters with only 1 compound: 6
# clusters with >5 compounds: 3
# clusters with >25 compounds: 2
# clusters with >100 compounds: 1
```

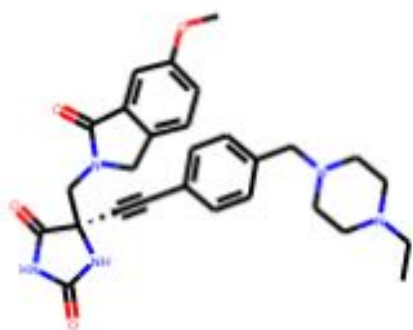
```
fig, ax = plt.subplots(figsize=(15, 4))
ax.set_xlabel("Cluster index")
ax.set_ylabel("Number of molecules")
ax.bar(range(1, len(clusters) + 1), [len(c) for c in clusters], lw=5);
```



Play with cut-off

```
print("Centroid molecules from first 7clusters:")  
# Draw molecules  
Draw.MolsToGridImage(  
    [compounds[clusters[i][0]][0] for i in range(7)],  
    legends=[compounds[clusters[i][0]][1] for i in range(7)],  
    molsPerRow=5,  
)
```

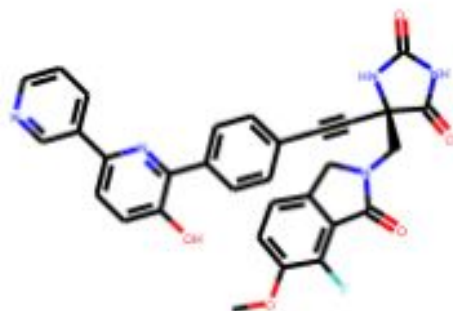
Centroid molecules from first 7clusters:



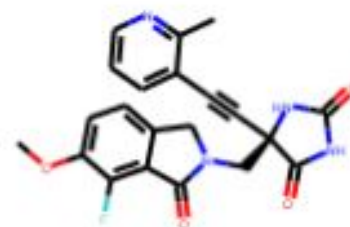
CHEMBL1287881



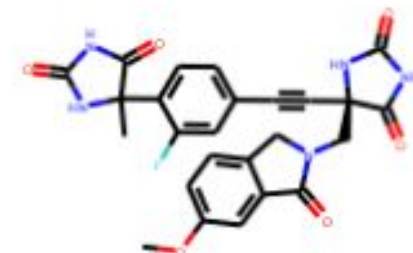
CHEMBL3947142



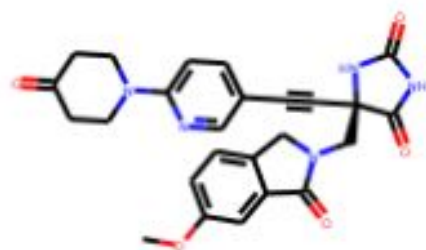
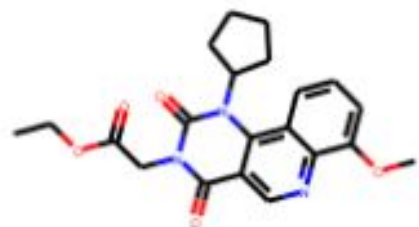
CHEMBL3640366



CHEMBL3642535



CHEMBL3642544



Спасибо за внимание