



**Vilniaus  
universitetas**

**t – SNE  
METODO PRISTATYMAS**

Ugnė Kniukštaitė, Simona Gelžinytė, Rugilė Bagdonaitė  
Duomenų mokslas, 3 kursas

# Turinys

1. t – SNE metodo pristatymas:

1. Pavadinimo iššifravimas ir pagrindinės algoritmo idėjos pristatymas.
2. Tinkami duomenys algoritmui.
3. Matematinis algoritmo veikimas.
4. „Žmogiškas“ algoritmo paaiškinimo veikimas su pavyzduku.
5. Hiperparametrai ir jų svarba.
6. Efektyvus algoritmo panaudojimas ir pastabos.
7. Palyginimas su PCA.
8. Realūs panaudojimo pavyzdžiai.
9. Privalumai ir trūkumai

2. Modifikacijos

3. Pritaikymas turimiems duomenims

---

# t – SNE METODAS

# Kas jis toks?

- t-SNE yra 2008 m. pristatytas netiesinis dimensijos mažinimo ir vizualizavimo metodas, kuris mažesnės dimensijos erdvėje siekia išsaugoti kiekvieno taško kaimynus (orientuotas į vidinės struktūros išsaugojimą).
- Algoritmas gali prisitaikyti prie pagrindinių duomenų ir skirtinguose regionuose atlikti skirtinges transformacijas.
- Jis neįtikėtinai lankstus, universalus ir patikimas algoritmas bei dažnai randantis struktūrą ten, kur kiti matmenų mažinimo algoritmai to padaryti negali.
- Gali išsaugoti globalias duomenų struktūras – klasterius, gerai apibrėžtus ir gana tiksliai atskirtus.
- Remiasi tikimybių pasiskirstymu su atsitiktiniu vaikščiojimu po kaimynystės grafus, kad būtų galima rasti duomenų struktūrą.



*Pagrindinis algoritmo tikslas:*  
tinkamai atvaizduoti artimus  
kaimynus mažesnėje dimensijoje



***t – SNE:*** t - distributed stochastic neighbor  
embedding arba Sjudento paskirstymo  
stochastinis kaimyninis įterpimas

# t-SNE algoritmo pavadinimo iššifravimas

Vilniaus  
universitetas

*Priešdėlis “t”* buvo įvestas siekiant atskirti algoritmą, kuriame taškų panašumui mažo matmens erdvėje apskaičiuoti naudojamas ne Gauso, o t pasiskirstymas.

t pasiskirstymas turi ilgesnes uodegas nei Gauso pasiskirstymas, o didėjant imties dydžiui šis skirstinys panašėja į normalųjį.

*Neighbour* (lt. kaimyninis) – labiausiai rūpinasi kaimyninių taškų atstumo išsaugojimu.

Be to algoritmas yra iš neprižiūrimo mokymo klasės, kur mašina stengiasi surasti tendencijas, modelį, nežinant, kokia išvestis iš tikrujų turi būti.

*Stochastic* (lt. stochasticinis) - tai reiškia procesą, kuriame dalyvauja duomenų imčių tikimybinis pasiskirstymas bei daugkartinis vykdymas su skirtinomis atsitiktinėmis sėklomis duos skirtinlus rezultatus.

*Embeding* (lt. įterpimas) – duomenų išdėstymas mažesnioje dimensijoje.

# Kokiems duomenims tinka?

Algoritmas ypač naudingas duomenims, kurių stebimų požymių ir klasės ryšys nėra tiesinis ir kuriuos prastai atspindi kiti matmenų mažinimo metodai.



K. Erdem, „t-SNE clearly explained“, 2020.

# Tinkami duomenys

- Lengviausiai šį algoritmą yra naudoti skaitiniams duomenims. Galima taikyti ir kitų tipų duomenims, tačiau tokiais atvejais reikia būti atsargiems ir įsitikinti, kad naudojate tinkamą atstumo metriką.
- Taip pat algoritmą reikia taikyti jau standartizuotiemis duomenims.

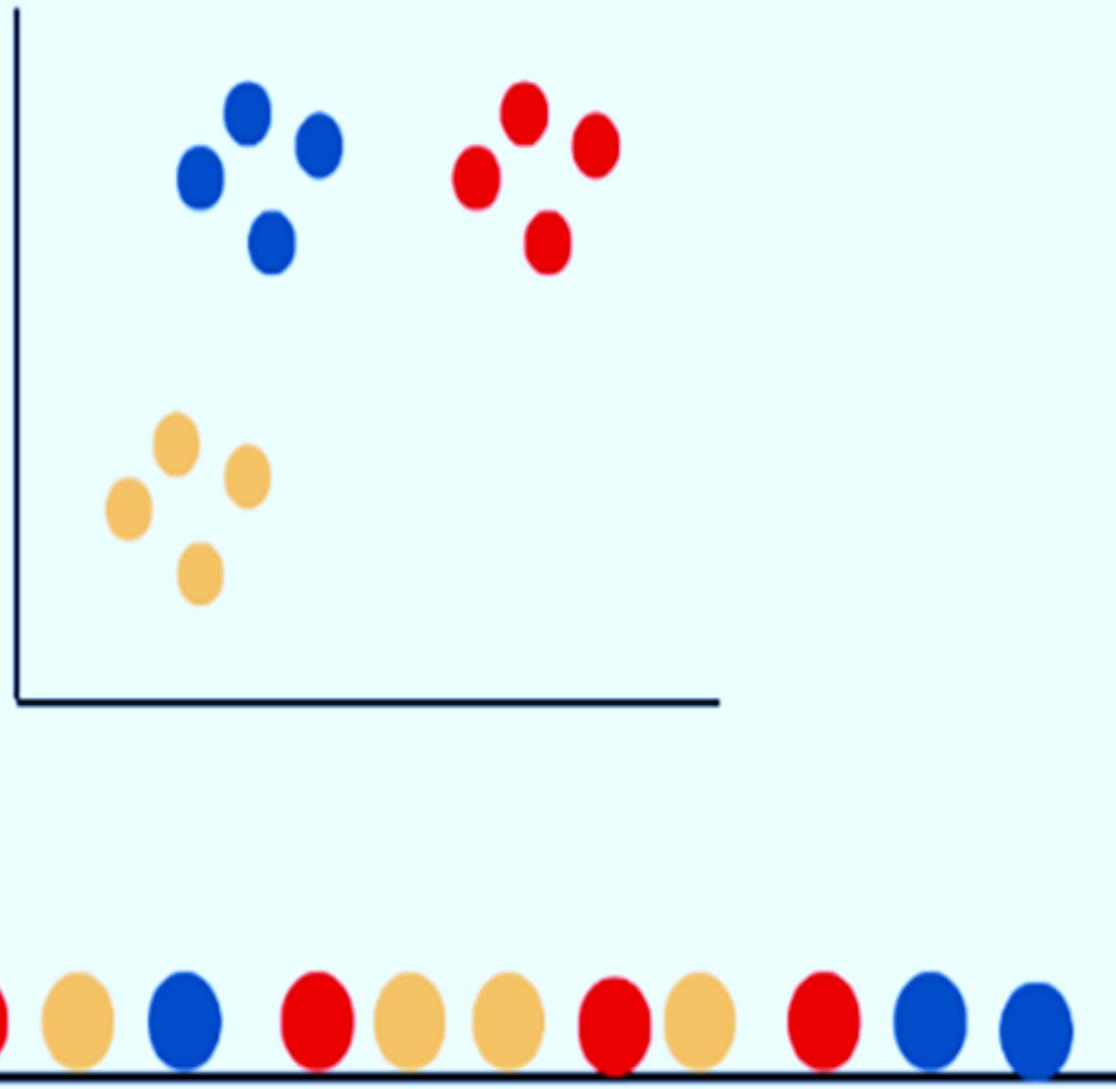
**Taigi, algoritmas tinka, kai nėra tiesinių sarysių bei kai turime didelę dimensiją.**

# Paprasčiausias algoritmo paaiškinimas

Pirma yra apskaičiuojamos tikimybės tarp taškų, kurios atspindi taškų panašumą. Taškai mažesniose dimensijos atidedami tol, kol pasiekiamama mažiausia Kulbako - Leiblerio divergencija. Ši divergencija gali būti laikoma atstumu, kaip vienas tikimybių pasiskirstymas skiriasi nuo kito, tai yra didesnės ir mažesnės dimensijos tikimybių pasiskirstymas.

## Klausimai, kurie gali iškilti:

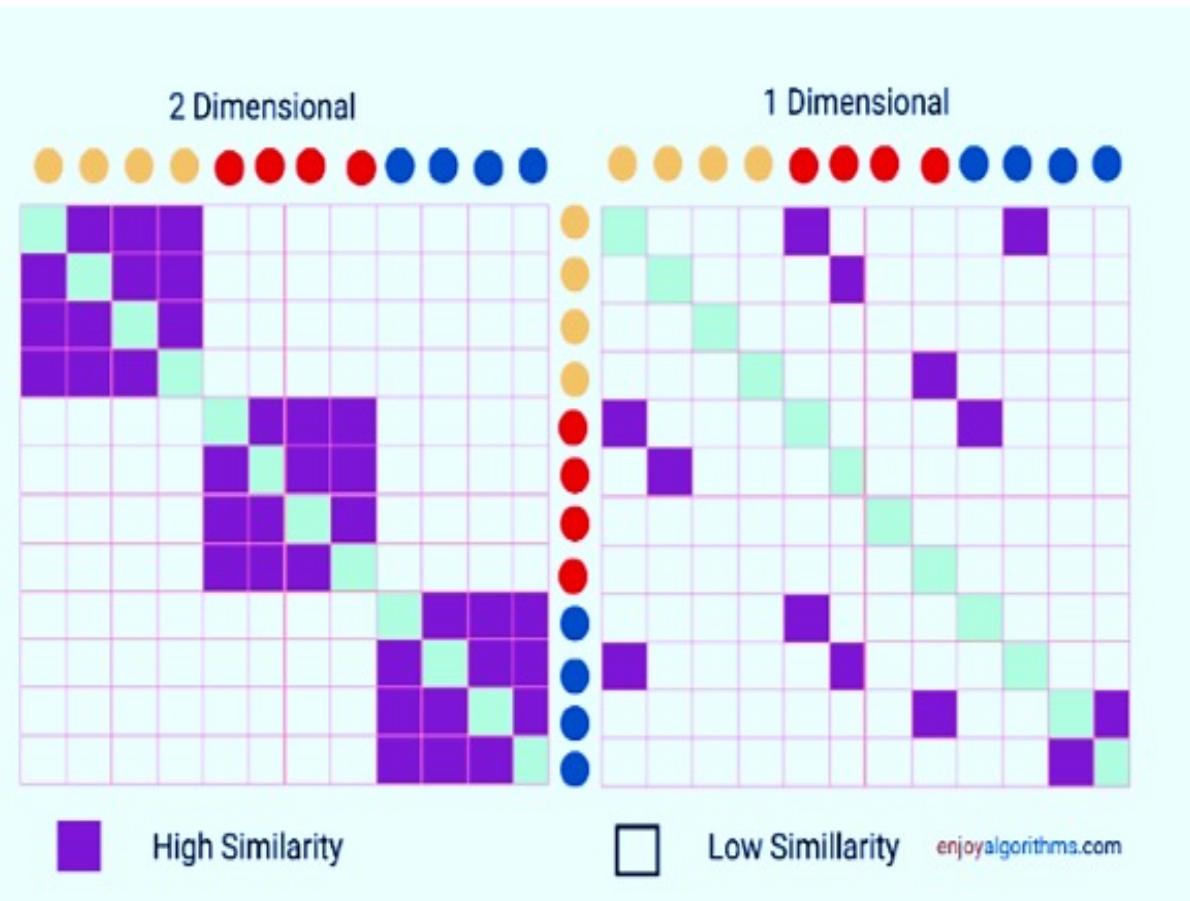
- Kaip apskaičiuojamos tikimybės?
- Kas yra Kulbako – Leiblerio divergencija?



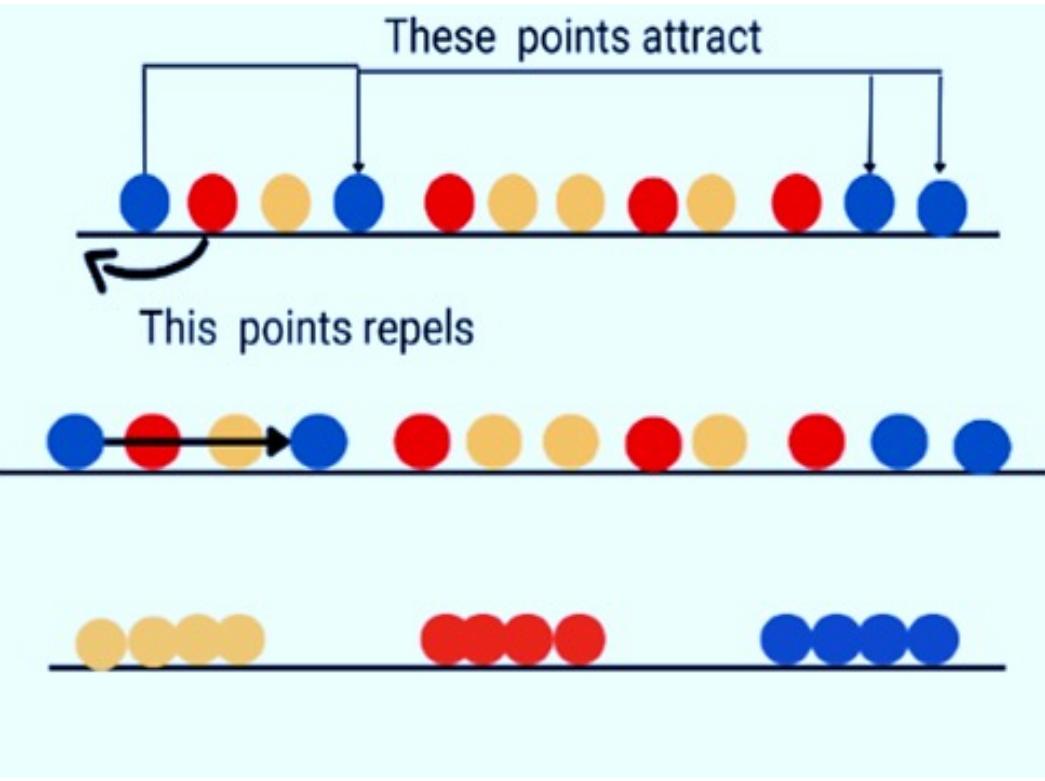
## Vaizdinis algoritmo paaiškinimas

Tarkime turime taškus 2D erdvėje ir juos norime suprojektuoti į 1D erdvę. Bet kaip turimus taškus atidedame į norimą dimensiją, jų padėti keisime atsižvelgiant į taškų panašumą.

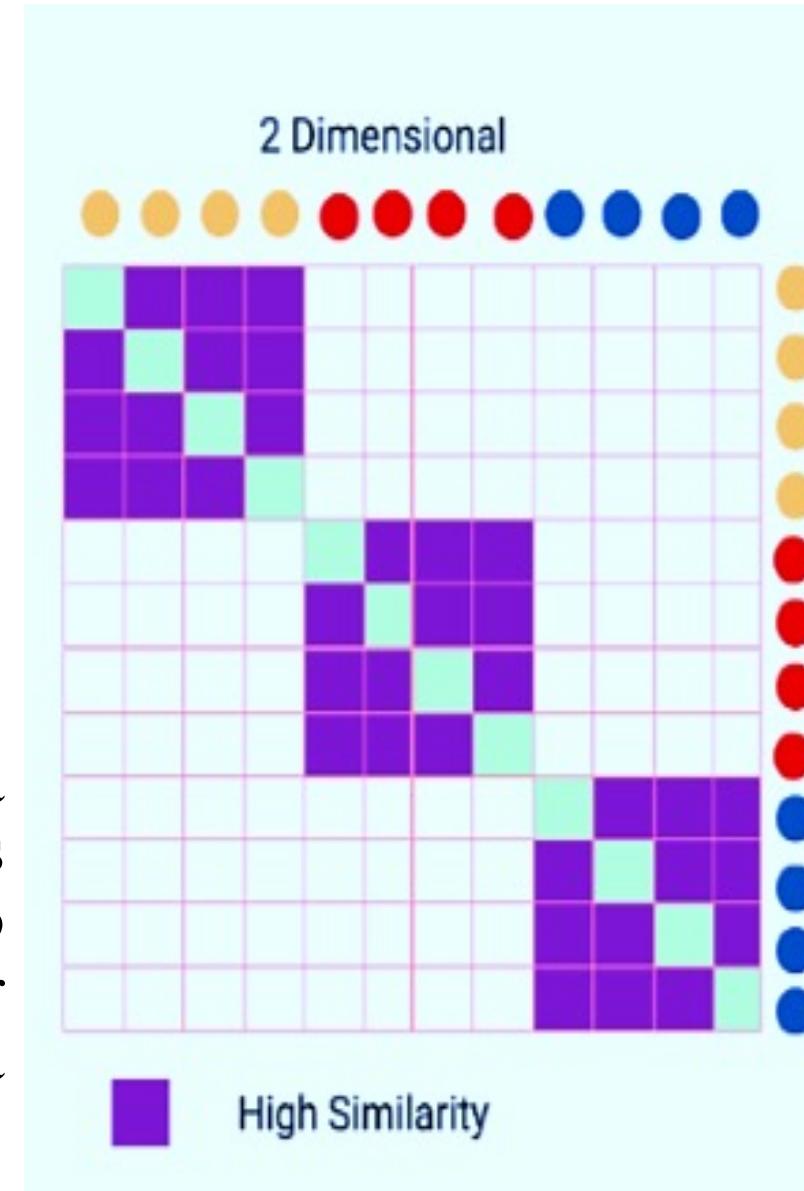
# Vaizdinis algoritmo paaiškinimas



H. Agarwal, „t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm“, enjoy algorithms,  
<https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>



Galime išsivaizduoti, jog taškus jungia įvairaus įtempimo spyruoklės. Stiprės spyruoklės jungia artimesnius taškus, o mažiau įtemptus tolimesnius kaimynus. Ir tada spyruoklės pradeda stumti arba pritraukti taškus



H. Agarwal, „t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm“, enjoy algorithms,  
<https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>

# Detalesnis algoritmo veikimas

1. Duomenims pradinėje dimensijoje suskaičiuojamos sąlyginės tikimybės atspindinčios panašumą, t. y. duomenų taško  $x_j$  panašumas su kitu tašku  $x_i$  yra sąlyginė tikimybė reiškianti, kokia tikimybė, kad taškas  $x_j$  pasirinks tašką  $x_i$  kaip kaimyną ir ši sąlyginė tikimybė apibrėžiama formule:

$$p_{i|j} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2 \times \sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2 \times \sigma_i^2}\right)}, \text{ čia } \|x_i - x_j\|^2 \text{ Euklidinis atstumas}$$

tarp taškų (galima naudoti ir kitą metriką),  $\sigma_i^2$  - dispersiją priklausanti nuo taško ir randama atliekant dvejetainę paiešką.

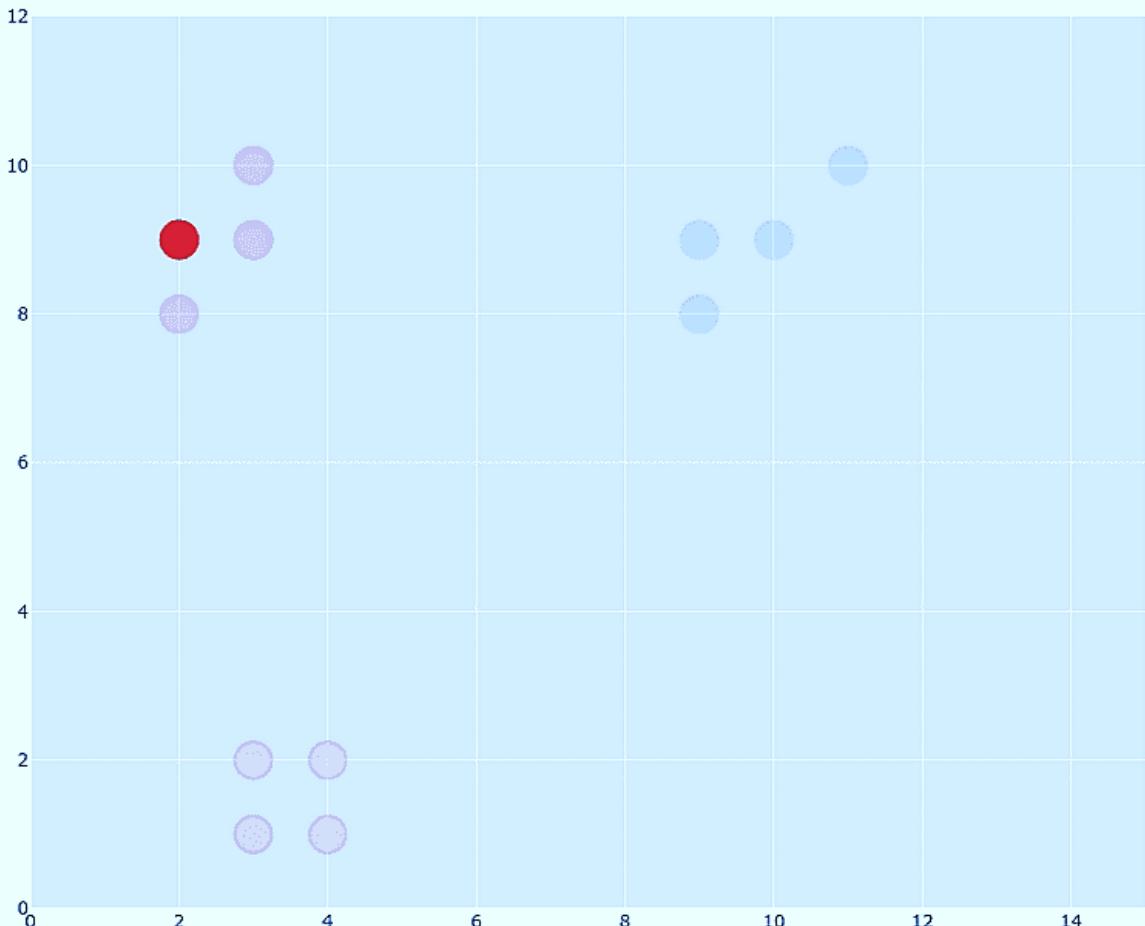
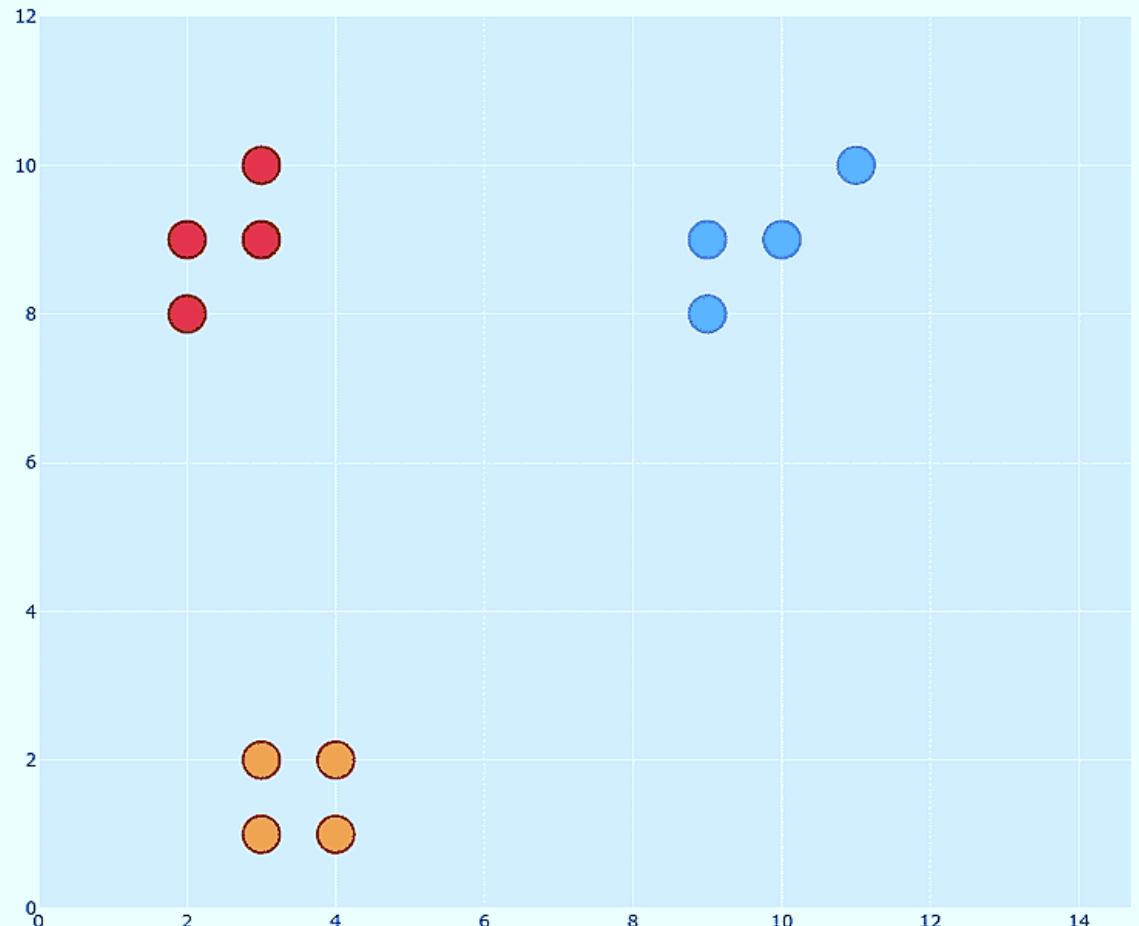
# Detalesnis algoritmo veikimas

Visų sąlyginių tikimybių suma lygi 1. Taip pat  $p_{i|i} = 0$ . Sukuriamas kiekvienam taškui jo kaimynų tikimybinis pasiskirstymas imant Gauso (normaluji) skirstinį, kurio centras ir yra  $x_i$  taškas. Taip pat mums dar reikia ir dispersijos kiekvienam taškui.

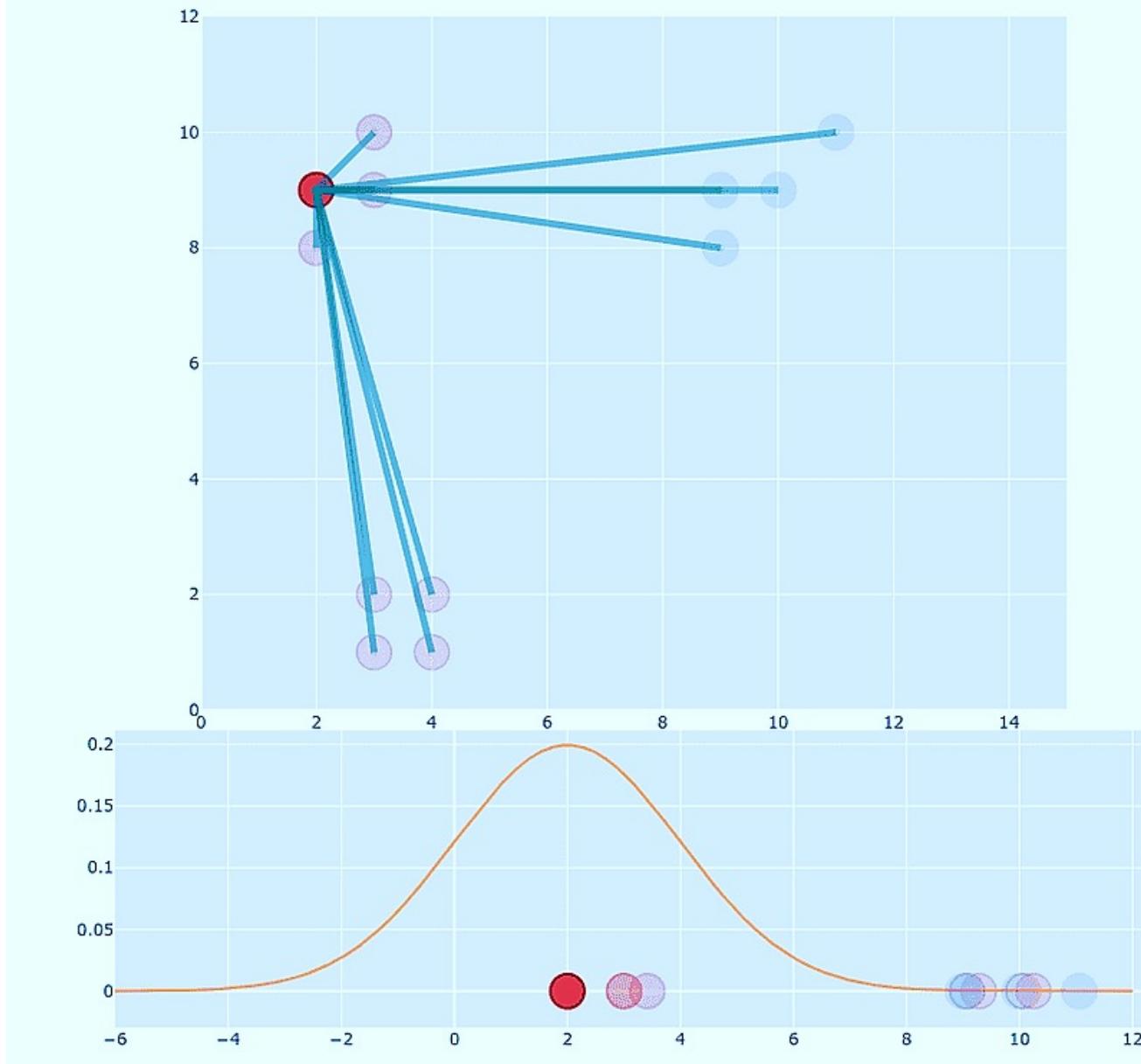
## Kodėl dispersija negali būti vienoda kiekvienam taškui?

Tikėtina, kad duomenų tankis gali skirtis, todėl mažesnio  $\sigma_i$  reikia tose vietose, kur tankis didesnis, o didesnio  $\sigma_i$  - ten, kur taškai yra toli. Kad gautume  $\sigma_i$ , turime atlikti tokią dvejetainę paiešką, kad Gauso pasiskirstymo, kurio centras yra  $x_i$ , kaimynų skaičius būtų lygus naudotojo nurodytam kaimynų skaičiui. Iš esmės, kuo didesnis kaimynų skaičius, tuo didesnė dispersijos vertė.

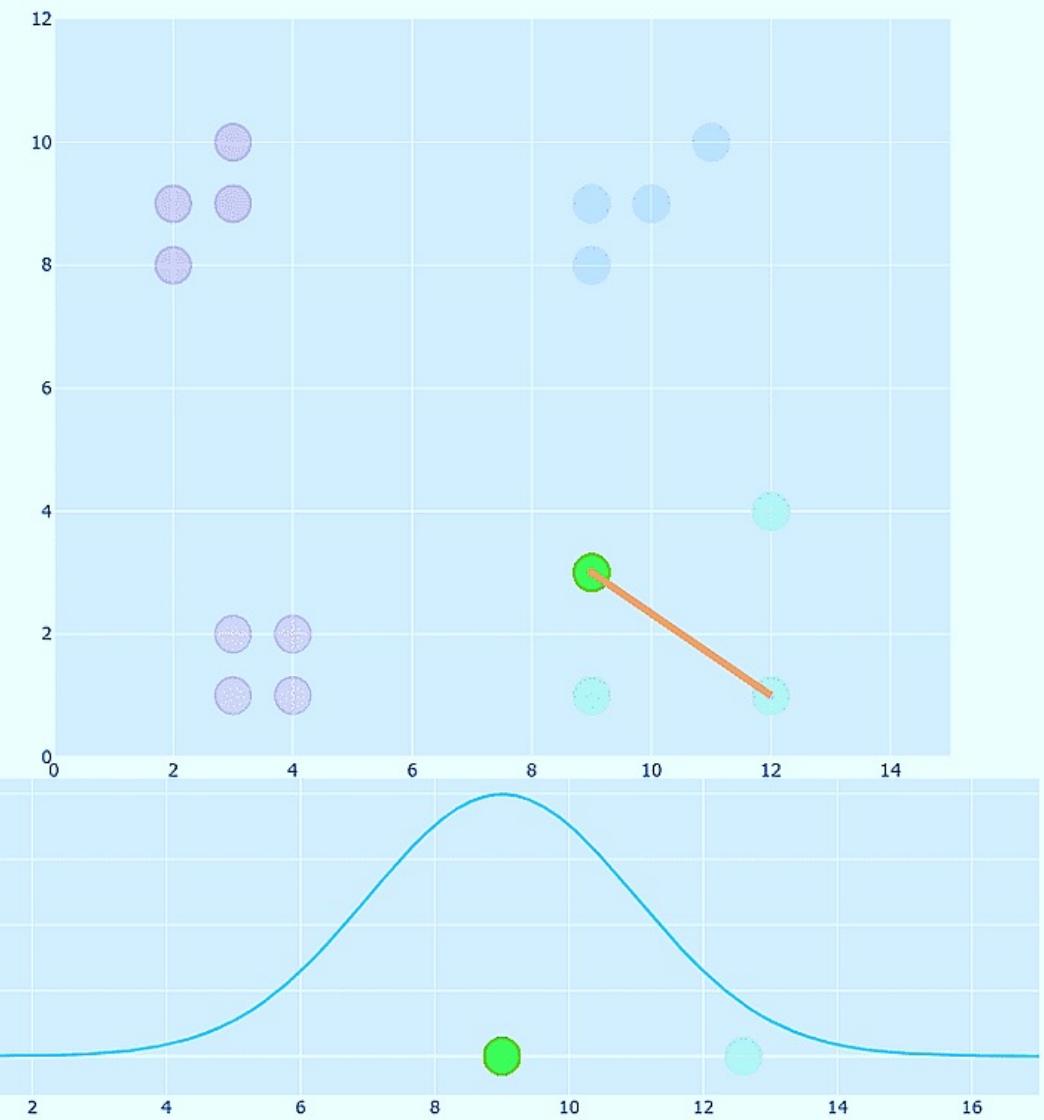
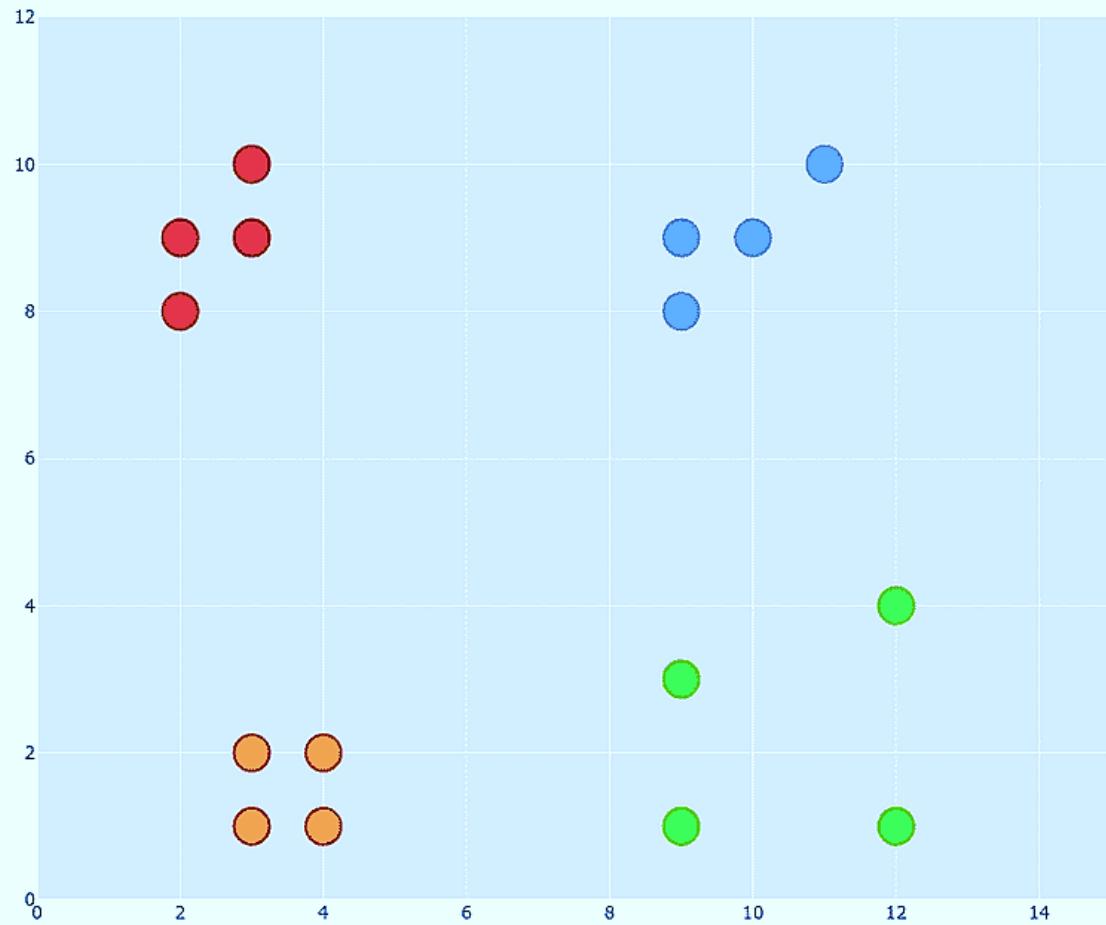
$$Perp(P_i) = 2^{-\sum p_{j|i} \log_2 p_{j|i}},$$
 čia laipsnio rodiklyje esantys reiškinys yra vadinimas entropija.



H. Agarwal, t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm, enjoy algorithms,  
<https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>

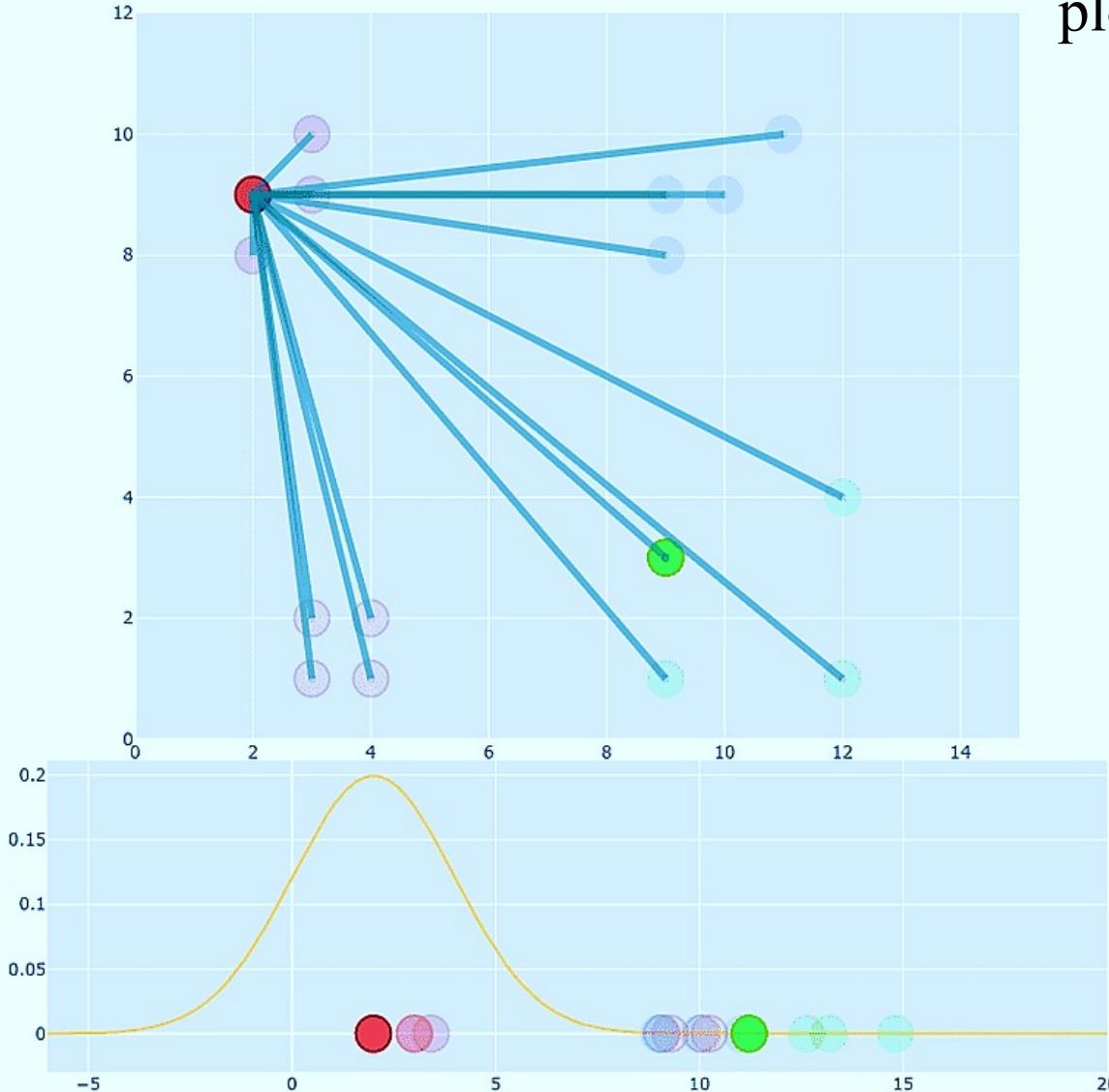


H. Agarwal, t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm, enjoy algorithms,  
<https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>

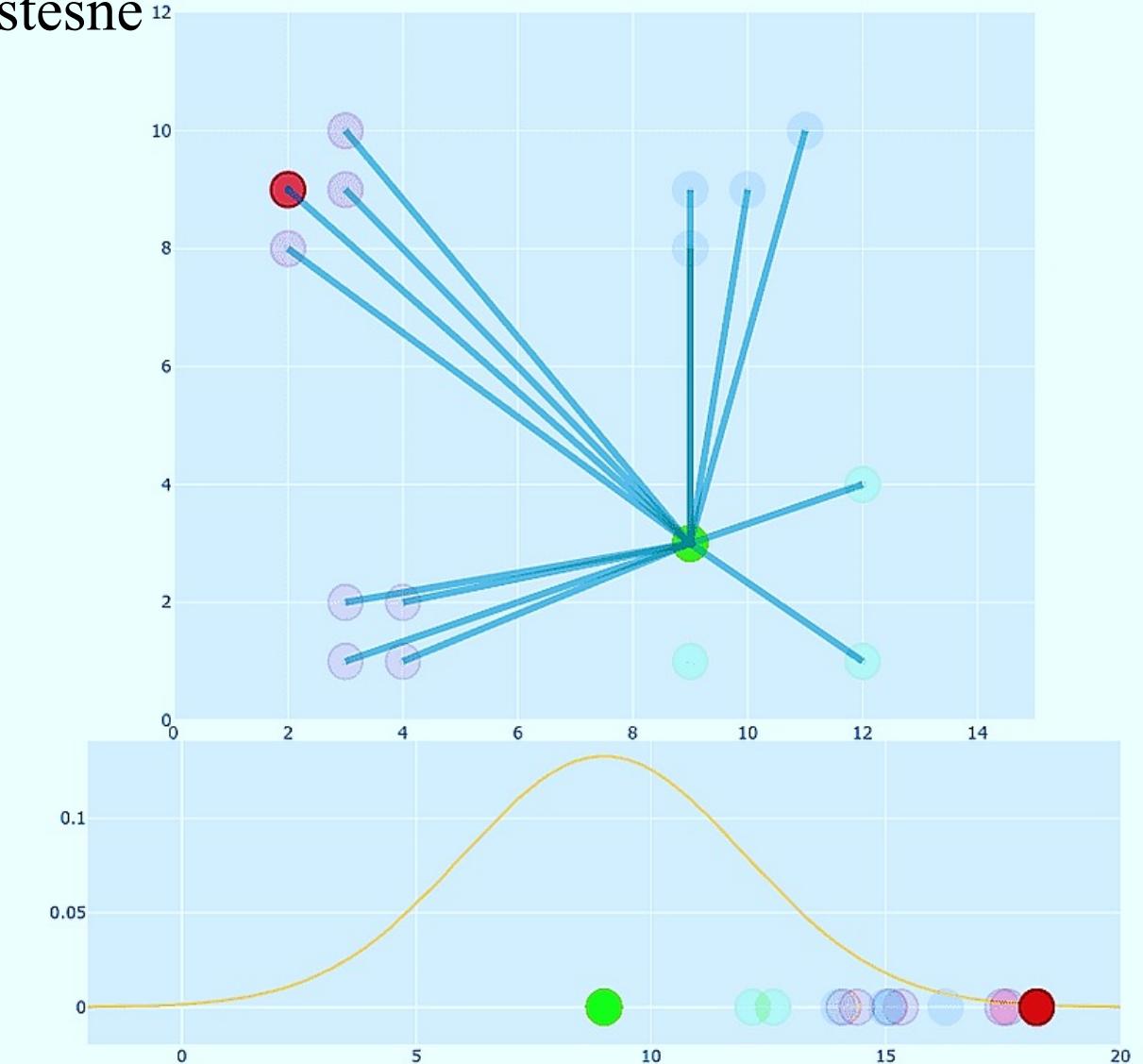


H. Agarwal, t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm, enjoy algorithms,  
<https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>

Taškui kaimynų skaičius didesnis



Kaimynų skaičius mažesnis, kreivė gavosi plokštesnė



# Detalesnis algoritmo veikimas

Preitoje skaidrėje gavome, jog  $p_{i|j}$  ir  $p_{j|i}$  tikimybės nėra lygios.

**Ką daryti tokiu atveju?**

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2 \times N}, \text{ kur } N - \text{dimensijų skaičius.}$$

# Detalesnis algoritmo veikimas

2. Atsitiktinai taškai atidedami mažesnėje dimensijoje ir jiems taip pat yra apskaičiuojamos sąlyginės tikimybės atspindinčios panašumą.

$$q_{i|j} = \frac{\exp\left(-\frac{\|x_i - x_j\|^2}{2 \times \sigma_i^2}\right)}{\sum_{k \neq i} \exp\left(-\frac{\|x_i - x_k\|^2}{2 \times \sigma_i^2}\right)}, \text{ čia } \|x_i - x_j\|^2 \text{ Euklidinis atstumas tarp taškų (galima naudoti ir kitą metriką), } \sigma_i^2 \text{ - dispersiją priklausanti nuo taško ir randama atliekant dvejetainę paiešką.}$$

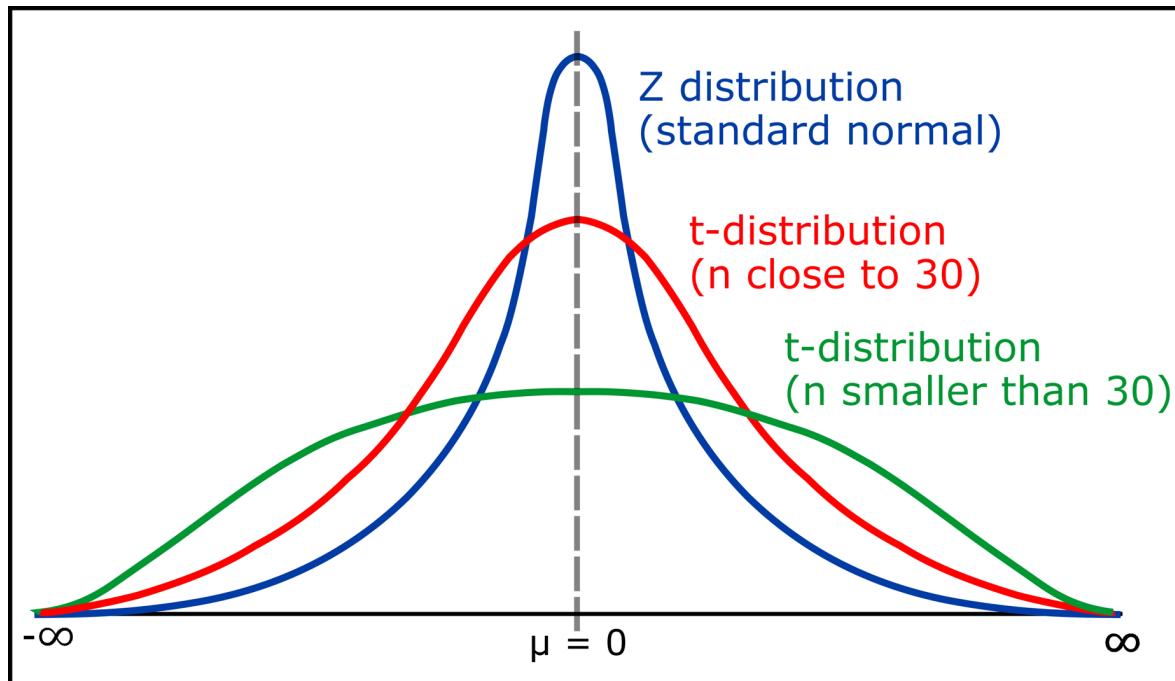
Dabar jau naudojamas Stjudento skirstinys su 1 laisvės laipsniu arba kitaip Koši skirstinys. Jis yra naudojamas norint panaikinti SNE algoritme atsiradusią perpildymo problemą.

# Perpildymo problema

- Perpildymo problema atsiranda tada, kai atstumas tarp klasterių yra didelis, palyginti su atstumu tarp klasterio taškų. Dėl to prarandama klasterio taškų skiriamoji geba.
- Perpildymo problema kyla dėl to, nes didesnės dimensijos kūnas paprastai yra talpesnis už mažesnės dimensijos kūną, todėl mažinant matmenis, taškai tampa perpildyti, tampa nebeįmanoma išsaugoti atstumus tarp taškų.
- Nėra pakankamai vietas visiems kaimynams sutalpinti.
- Tai viena didžiausių SNE problemų.

# Perpildymo problema - sprendimas

Sumažintoje dimensijoje sąlygines tikimybes skaičiuoti pasitelkiama naudojant Stjudento skirstinį ne normalujį, nes t skirstinys turi sunkesnę uodegą, kurioje geriau gali sutilpti taškai.



# Kuo skiriasi t – SNE nuo SNE?

- Sumažintoje dimensijoje sąlyginės tikimybės yra skaiciuojamos naudojant t skirstinį (t – SNE), o SNE algoritme, naudojamas normalusis skirstinys.
- Pradinėje dimensijoje yra naudojamas simetrinis tikimybių skirstinys, toks, kad  $p_{ij} = p_{ji}$  (t – SNE), o SNE algoritme naudojamas asimetrinis.

# Gautos projekcijos įvertinimas

Norint įvertinti, kaip gerai yra projekcija yra atidėta sumažintoje erdvėje, yra skaičiuojama Kulbako – Leiblerio divergencija – sąnaudų funkcija:

$C = \sum_i \sum_{j \neq i} p_{ij} \times \log \left( \frac{p_{ij}}{q_{ij}} \right)$ , kur  $p_{ij}$  ir  $q_{ij}$  yra porų tikimybės pradinėje ir sumažintoje dimensijoje atitinkamai.

Vietinę struktūrą galima tiksliau nustatyti dėl to, kad sąnaudų funkcija yra proporcinga porinei tikimybei didelio matmens erdvėje. Santykinai nutolę taškai turi daug mažesnį poveikį sąnaudų funkcijai.

Idealiu atveju divergencija turėtų būti lygi 0, t. y. pradinėje ir sumažintoje dimensijoje visų taškų atstumai yra išsaugoti.

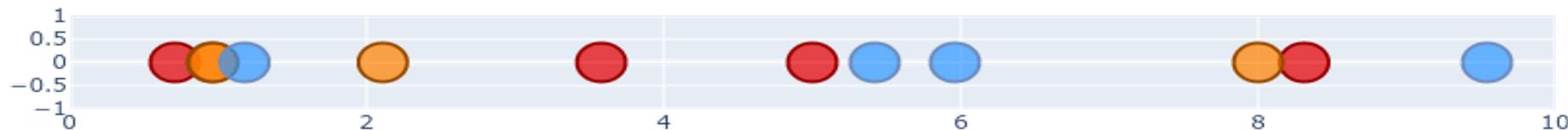
# Sprendinio optimizavimas

Norėdami sumažinti šį rezultatą, atliekame gradientinį nusileidimą per nustatyta iteracijų skaičių:

$$\frac{\delta C}{\delta y_i} = 4 \times \sum_j (p_{ij} - q_{ij}) \times (y_i - y_j) \times \left(1 + \|y_i - y_j\|^2\right)^{-1}, \text{ čia } y_i \text{ ir } y_j \\ \text{taškų koordinatės mažesnioje dimensijoje, } \|y_i - y_j\|^2 \text{ - Euklidinis} \\ \text{atstumas tarp jų.}$$

# Gradiento interpretacija

Fiziškai gradientą galima interpretuoti kaip rezultatinę jėgą, kurią sukuria spyruoklių rinkinys tarp taško  $y_i$  ir  $y_j$ . Spyruoklė atstumia arba pritraukia taškus, priklausomai nuo to, ar atstumas tarp jų projekcijoje yra per mažas, ar per didelis, kad atspindėtų dviejų pradinės dimensijos duomenų taškų panašumą. Jėga, kurią veikia spyruoklė tarp  $y_i$  ir  $y_j$  yra proporcinga skirtumo ilgiui, taip pat proporcinga skirtumo standumui, kuris yra nesutapimas ( $p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j}$ ).



# Algoritmo veikimo pastabos

- Išterpiant duomenų rinkinį į mažesnę dimensiją, gali atsirasti dviejų rūšių klaidos, artimiausi kaimynai atvaizduojami kaip tolimi taškai ( $p_{ij}$  yra didelis, o  $q_{ij}$  mažas), o taškai, kurie yra toli, atvaizduojami kaip kaimynai ( $p_{ij}$  yra mažas, o  $q_{ij}$  didelis).
- Matujant porinius atstumus didelio ir mažo matmenų erdvėse, naudojant skirtingus tikimybių pasiskirstymus, galima optimaliau vizualizuoti klasterio viduje esančias detales.
- Kad būtų išvengta ankstyvojo klasterizavimo, t-SNE ankstyvuosiuose etapuose į sąnaudų funkciją įtraukia L2 baudą. Ją galima laikyti standartiniu reguliavimu, nes ji leidžia algoritmui nesusikoncentruoti į vietines grupes.

# Reguliuojami parametrai

## *Iteracijų skaičius*

Paprastai didelis iteracijų skaičius leidžia algoritmui konverguoti su mažesniais nuostoliais. Jei gautas grafikas su keistomis suspaustomis formomis, tikėtina, kad procesas buvo sustabdytas per anksti. Svarbu turėti tiek iteracijų, kad būtų pasiekta stabili konfigūracija.

## *Perpleksišumas / Kaimynų skaičius*

Tai artimų kaimynų skaičius kiekvienam taškui, kuris taip pat nurodo, kaip subalansuoti dėmesį tarp vietinių ir globalių duomenų aspektų.

Iš esmės keičia Gauso plotį taškų poros tikimybėms apskaičiuoti. Didesnis perpleksišumas padidins tolimesnių duomenų svorių sumažintoje dimensijoje.

## *Mokymosi greitis*

Skaliaras, nuo kurio priklauso įterptujų verčių atnaujinimo mastas kiekvienoje iteracijoje. Pasirinkus didesnį mokymosi greitį, paprastai sprendinys konverguos greičiau, tačiau jei jis bus per didelis, įterptosios reikšmės gali nekonverguoti ir susidaryti lygiareikšmių taškų kamuolys.

# Reguliuojami parametrai

## *random\_state*

Galima nurodyti šio parametro reikšmę, norint išvengti skirtinį rezultatų gavimo, naudojant identiškus parametrus.

## *Atstumo metrika*

Pagal nutylėjimą yra naudojama Euklidinė metrika. Tačiau ji taip pat gali būti: „chebyshev“, „cityblock“, „correlation“, „cosine“, „hamming“, „jaccard“, „mahalanobis“, „minkowski“, „seuclidean“, „squared\_euclidean“, „fasteuclidean“, „fastseuclidean“, „spearman“, „canberra“, „braycurtis“, „dice“, „kulcsinski“, „matching“, „rogerstanimoto“, „rusellrao“, „sokalmichener“, „sokalsneath“, „yule“. Ji yra svarbus pasirinkimas.

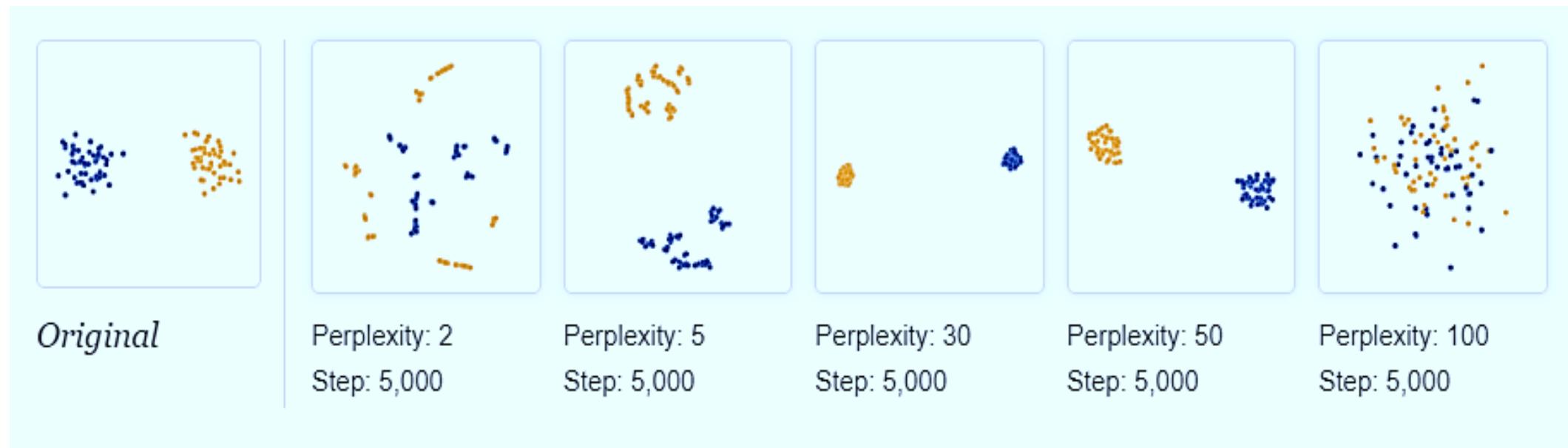


Svarbiausi parametrai yra *iteracijų*  
ir *kaimynų skaičius*



# Kaimynų skaičiaus svarba

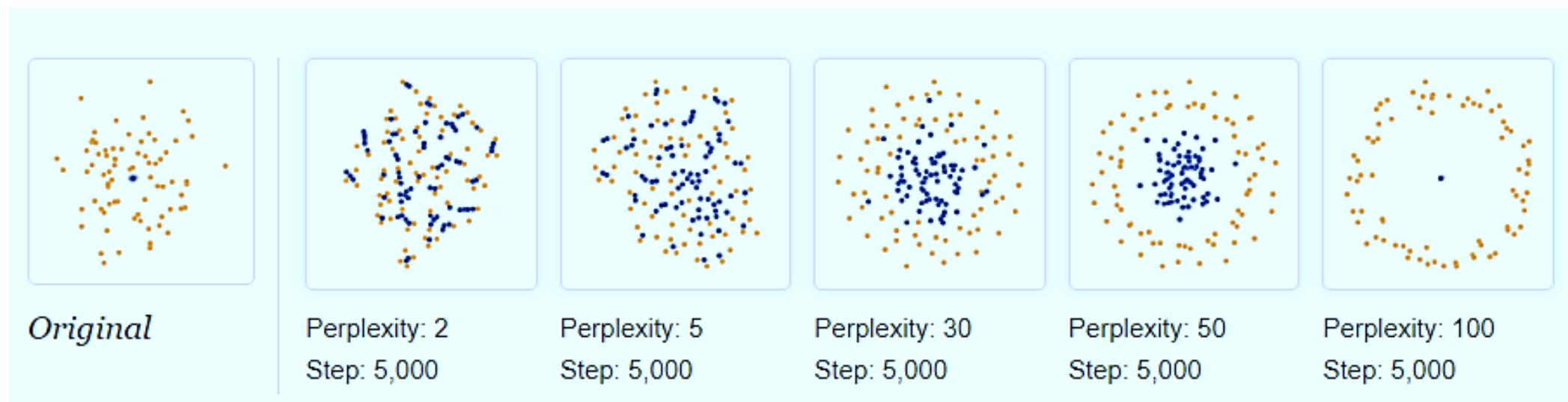
Fiksavus iteracijų skaičių 5000, galime matyti, jog geriausiai duomenų struktūra išsaugo, kai kaimynų skaičius yra 50.



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

# Kaimynų skaičiaus svarba

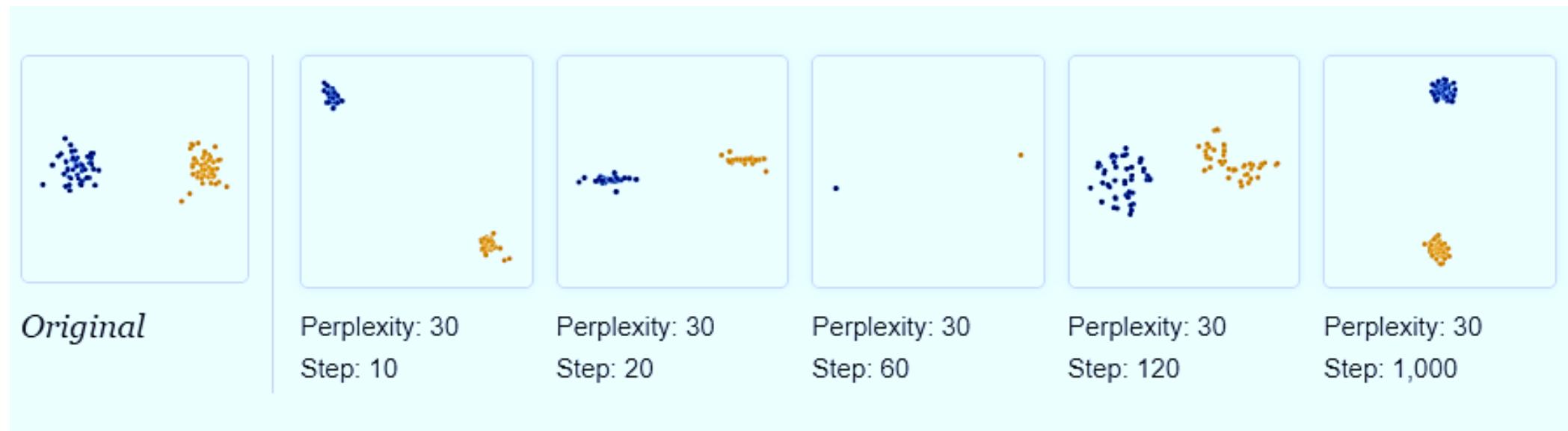
Kai kaimynų skaičius 30, tai algoritmas teisingai parodo pagrindinę topologiją, tačiau t-SNE gerokai pervertina mažesnės taškų grupės dydį. Esant 50 kaimynų skaičiui, pastebimas naujas reiškinys: išorinė grupė tampa apskritimu - brėžinyje bandoma pavaizduoti, kad visi jos taškai yra maždaug vienodai nutolę nuo vidinės grupės.



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

# Iteracijų skaičiaus svarba

Fiksavus kaimynų skaičių 30, matome, jog geriausiai grupės topologija atspindi, kai iteracijų skaičius yra 120.





t-SNE neįtikėtinai lankstus ir dažnai gali rasti struktūrą ten, kur kiti matmenų mažinimo algoritmai to padaryti negali. Deja, dėl šio lankstumo jį sudėtinga interpretuoti.





Būtina išsibandyti keletą parametro  
derinių, prieš pereinant į rezultatų  
analizavimą.



# Patarimai efektyviam algoritmo panaudojimui

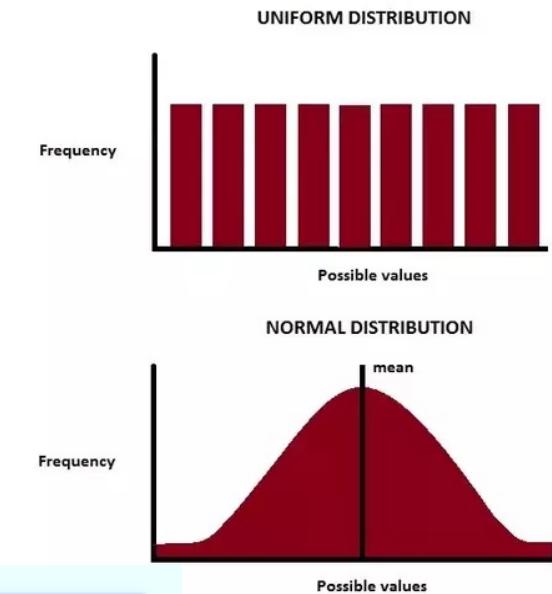
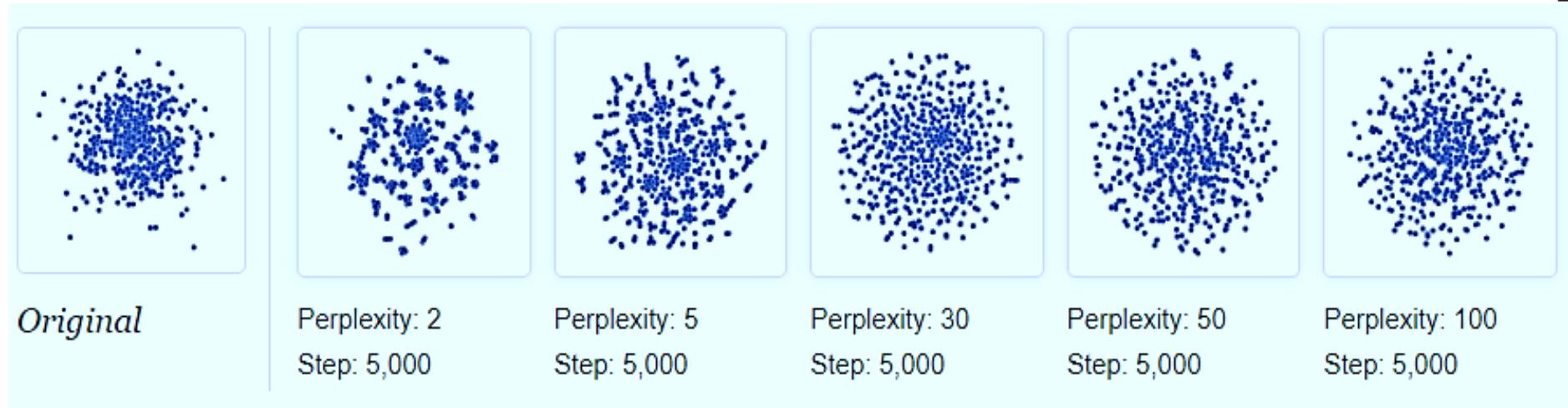
- Kadangi algoritmas yra stochastinis, kiekvienas bandymas gali pateikti skirtinę rezultatą. Norint šito išvengti galima nurodyti konkrečią reikšmę *random\_state* parametru visiems bandymams.
- Kad t-SNE būtų prasmingas, turime pasirinkti tinkamą kaimynų skaičių. Šis rodiklis subalansuoja vietinius ir globalius duomenų rinkinio aspektus. Labai didelė reikšmė lems klasterių susijungimą į vieną didelį klasterį, o maža reikšmė sukurs daug artimų mažų klasterių, kurie bus beprasmiai. Siūloma reikšmė yra nuo 5 iki 50.

# Patarimai efektyviam algoritmo panaudojimui

- Klasterių dydžiai bet kuriame t-SNE grafike neturi būti vertinami pagal standartinį nuokrypi, dispersiją ar kitus panašius rodiklius. Taip yra todėl, kad t-SNE išplečia tankesnius klasterius ir sutraukia retesnius klasterius, kad išlygintų klasterių dydžius. Tai yra viena iš priežasčių, dėl kurių gaunami ryškūs ir aiškūs grafikai.
- Modelių galima rasti ir atsitiktiniame triukšme, todėl prieš nusprendžiant, ar duomenyse yra modelis, reikia patikrinti kelias algoritmo realizacijas su skirtingais hiperparametru rinkiniais.
- Norint interpretuoti gautos vizualizacijos topologiją, ją reikėtų vertinti bent jau iš kelių grafikų.

# Modelis atsirandantis dėl triukšmo

Šiam tyrimui atlikti buvo panaudoti atsitiktinai sugeneruoti duomenys iš normaliojo skirstinio. Matome, kai kaimynų skaičius lygus 2, atsiranda klasteriai. Esant kaimynų skaičiui 30, duomenys panašėja iš tolygiojo skirstinio.



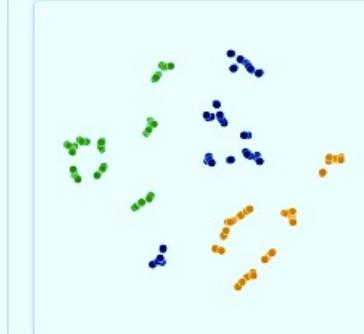
# Atstumai tarp klasterių

Nerekomenduojama daryti išvadų remiantis vien atstumu tarp klasterių.

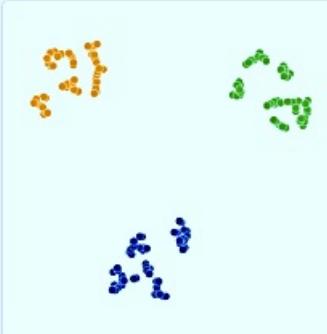
Klasteriuose taškų skaičius 50



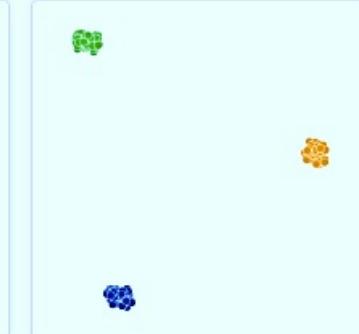
Original



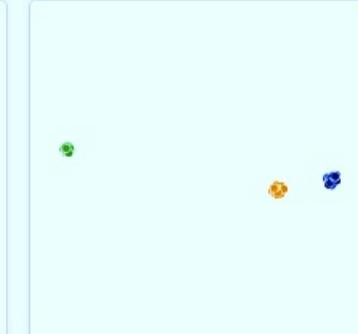
Perplexity: 2  
Step: 5,000



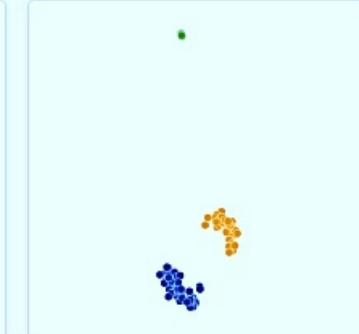
Perplexity: 5  
Step: 5,000



Perplexity: 30  
Step: 5,000



Perplexity: 50  
Step: 5,000

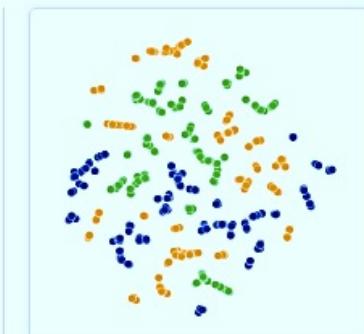


Perplexity: 100  
Step: 5,000

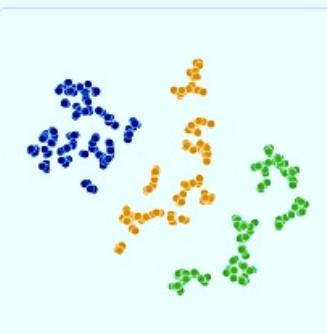
Klasteriuose taškų skaičius 200



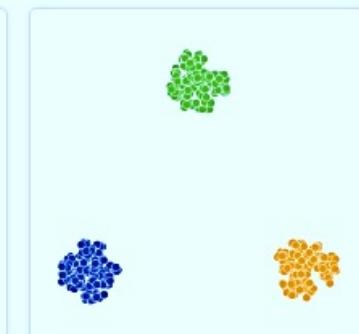
Original



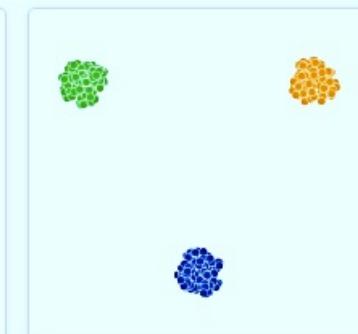
Perplexity: 2  
Step: 5,000



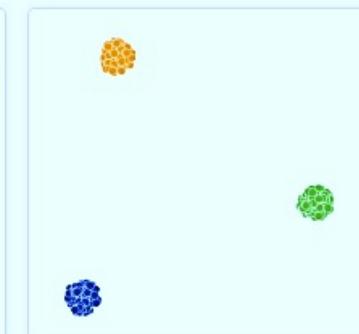
Perplexity: 5  
Step: 5,000



Perplexity: 30  
Step: 5,000



Perplexity: 50  
Step: 5,000

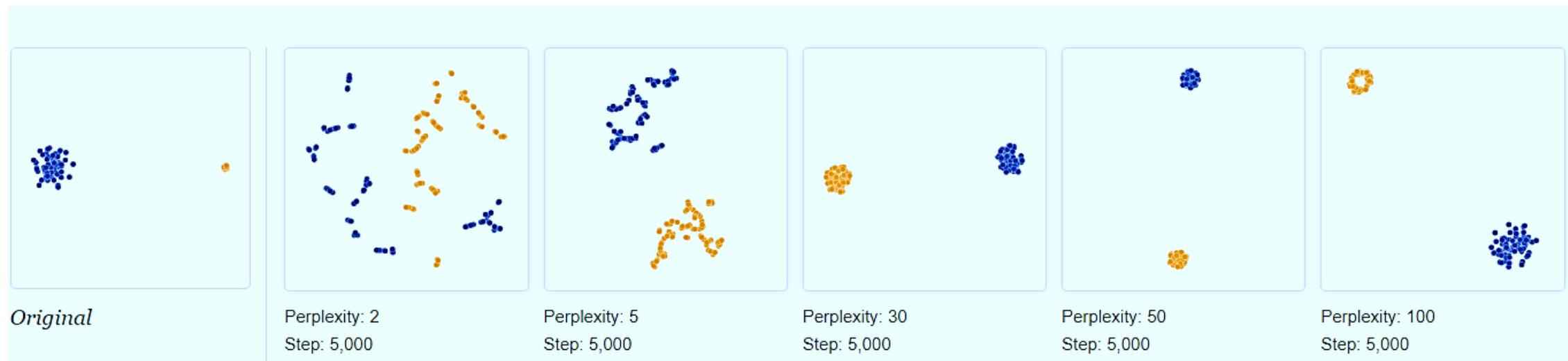


Perplexity: 100  
Step: 5,000

Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

# Klasterių dydžiai

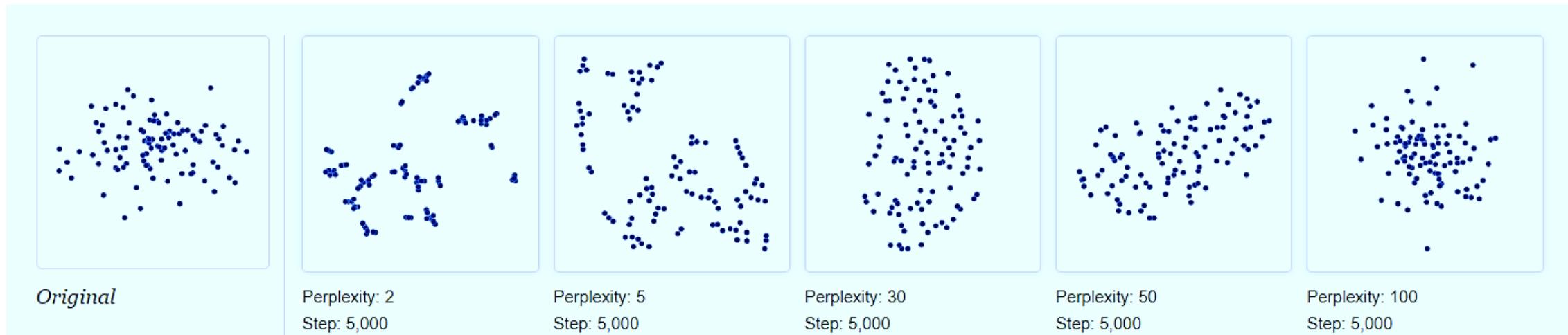
Šis algoritmas susiaurina plačiai išsibarsčiusius duomenis ir išplečia tankiai supakuotus. t-SNE diagramoje negalima matyti santykinių klasterių dydžių.



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

# Matomos formos

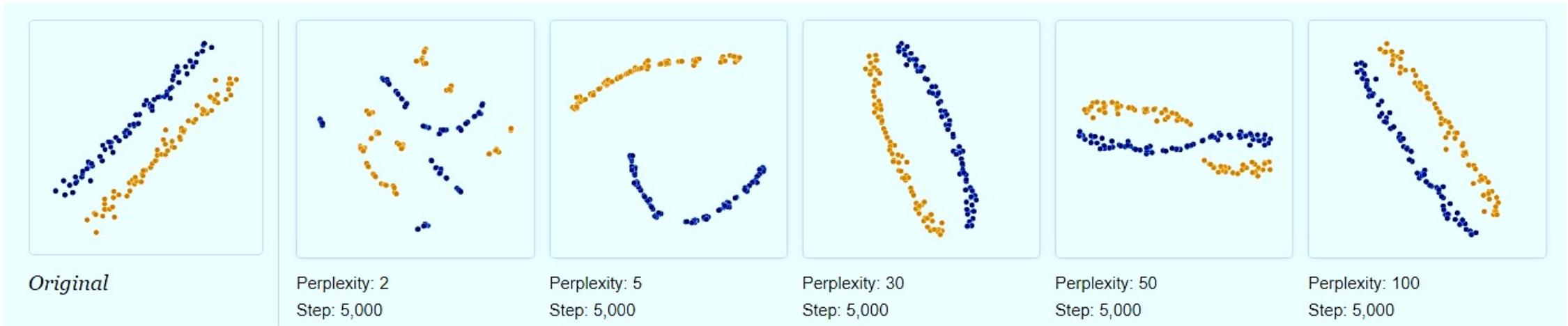
Yra duoti duomenys, kuriuose turėtų atsispindėti elipsoidinė struktūra. Kai kaimynų skaičius 2 ir 5, pagrindinė perteikiama informacija yra klasteriai. Kai kaimynų skaičių padidiname, atsiskleidžia turima elipsoidinė forma.



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

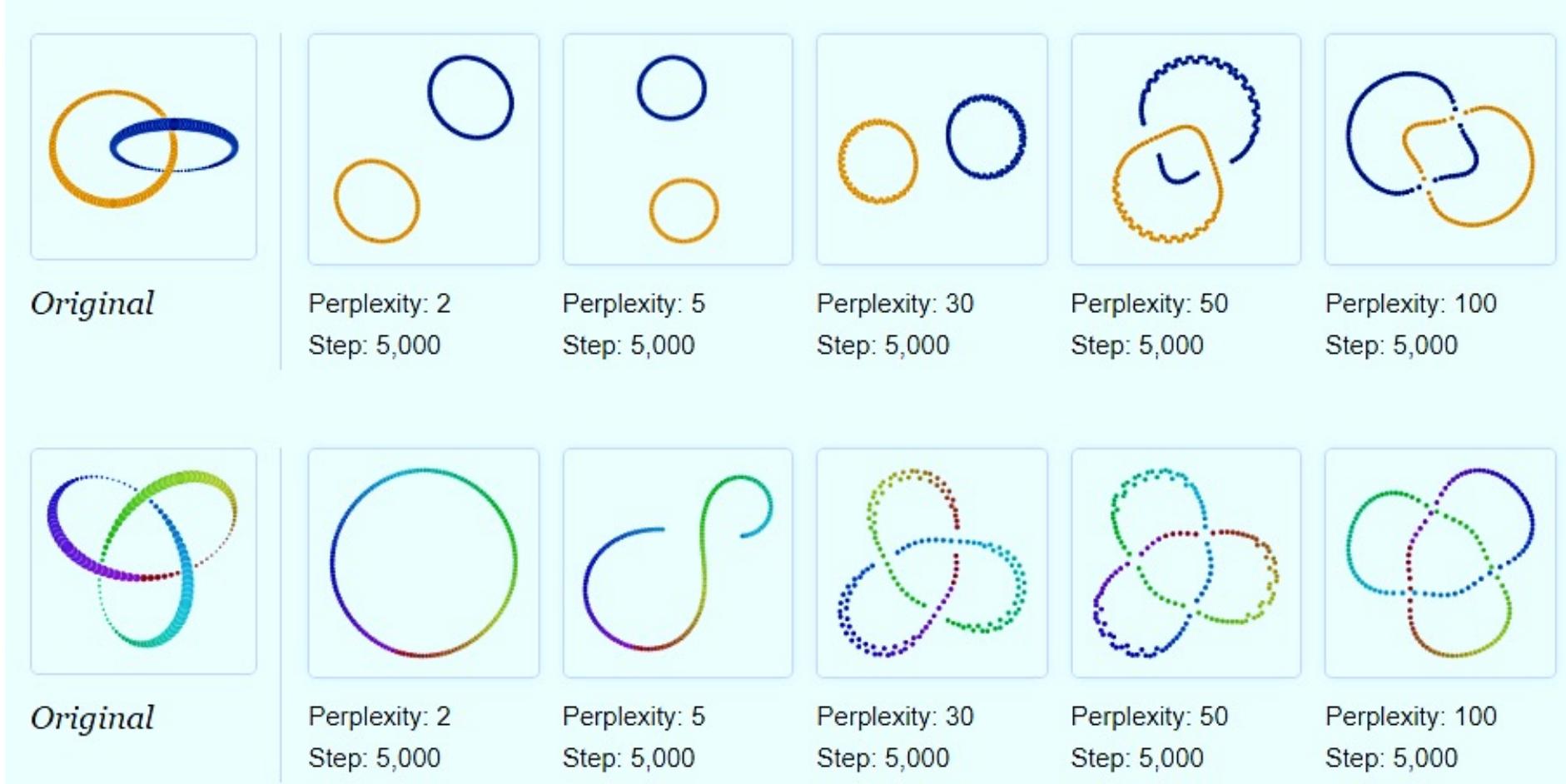
# Matomos formos

Net ir geriausiais atvejais pastebimas subtilus iškraipymas (kaimynų skaičius 30): t-SNE diagramoje linijos šiek tiek išlenktos į išorę. Priežastis ta, kad, kaip įprasta, t-SNE linkės išplėsti tankesnius duomenų regionus. Kadangi klasterių viduriuose yra mažiau tuščios erdvės nei jų galuose, algoritmas juos padidina.



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>

# Duomenų struktūra



Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016.  
<http://doi.org/10.23915/distill.00002>

Vaizduojami  
trimačiai  
duomenys ir  
algoritmo  
gebėjimas  
juos  
atvaizduoti  
mažesnėje  
dimensijoje.

# Pastabos

Atstumai tarp klasterių gali keistis, nes globalioji geometrija glaudžiai susijusi su optimaliu kaimynų skaičiumi. O duomenų rinkinyje, kuriame yra daug klasterių su skirtingu elementų skaičiumi, viena reikšmė negali optimizuoti atstumų visiems klasteriams.

# Palyginimas su PCA

Vilniaus  
universitetas

	t-SNE	PCA
<b>Metodas</b>	Stochastinis	Deterministinis - visada grąžins tą patį atsakymą su vienodomis parametru reikšmėmis
<b>Išsaugota globali struktūra</b>	Siekia išsaugoti lokalią struktūrą	Taip
<b>Pritaikymas naujiems duomenims</b>	Ne	Taip
<b>Trūkstamos reikšmės</b>	Netinka	Netinka, nebent tikimybinis pca
<b>Greitis</b>	Lėtas	Greitas
<b>Netiesiniai sąryšiai</b>	Atvaizduoja	Nepagauna

# Palyginimas su PCA

- t-SNE yra iteracinis metodas, todėl, kitaip nei PCA, jo negalima taikyti kitam duomenų rinkiniui. PCA duomenims mažinti naudoja globalią kovariacijos matricą. Šią matricą galima gauti ir taikyti naujam duomenų rinkiniui. Tai naudinga, kai reikia pabandyti sumažinti požymių sąrašą ir dar kartą panaudoti matricą, sukurtą iš mokymo duomenų. Šis procesas nepavyks su t – SNE: papildomų duomenų į projekciją negalima lengvai įtraukti iš naujo neperskaičiavus.
- PCA tikslas padengti kuo daugiau dispersijos, o t - SNE išlaikyti kuo panašesnius ryšius tarp kaimynų

# Algoritmo panaudojimas

- Naudojamas dirbant su konvoliuciniais tinklais.
- Nors t-SNE kūrėjas paminėjo jo naudojimo atvejus tokiose srityse kaip klimato tyrimai, kompiuterių saugumas, bioinformatika, vėžio tyrimai ir t. t.. Pritaikius šią metodą, jo rezultatus galime naudoti įvairiuose prižiūrimo modeliavimo procesuose.
- Jis plačiai taikomas vaizdų apdorojimo, genominių duomenų ir kalbos apdorojimo srityse. Jis buvo panaudotas smegenų ir širdies skenavimo analizei pagerinti.

# Algoritmo panaudojimas

Šis algoritmas nelabai tinka klasifikavimui, tačiau jo išvestį galima naudoti kaip kitų klasifikavimo algoritmų įvesties požymį. Vienas detalesnis bandymas - veido atpažinimas pagal Japonijos moterų veido išraiškos duomenų bazę (JAFFE) naudojant t-SNE ir AdaBoostM2. Eksperimentų rezultatai parodė, kad naudota kombinacija, pasiekė geresnių rezultatų, palyginti su tradiciniais algoritmais, tokiais kaip PCA, LLE ir SNE.

Dimensijos mažinimo algoritmas	PCA	LLE	SNE	t - SNE
Gautas tikslumas panaudojus AdaboostM2 klasifikavimo algoritmą	75,4 %	87,7 %	90,6 %	94,5 %

## Privalumai

- Išsaugo vietines duomenų kaimynystes. Stebėjimai, kurie yra arti vienas kito įvesties požymių erdvėje, turėtų būti arti vienas kito ir transformuotoje požymių erdvėje.
- Taip pat dažniausiai išsaugo ir globalią duomenų struktūrą
- Tinka duomenims, kuriuose yra netiesiniai sąryšiai.
- Nejautrus išskirtims.

## Trūkumai

- Reikalauja daug skaičiavimo resursų, todėl yra gana lėtas.
- Turi hiperparametrus, kuriuos reikia tinkamai sureguliuoti, kad modelis gerai veiktu.
- Netinka duomenims su praleistomis reikšmėmis.
- Nedeterministinis - negausite lygiai tokios pačios įvesties, naudojant tuos pačius parametru rinkinius, nors rezultatai greičiausiai bus panašūs.
- Atsiradęs triukšmas, gali parodyti neegzistuojančius modelius.
- Gali sukurti netikrus klasterius, dėl duomenų ištempimo.
- Jautrus masteliui, dėl to duomenys turi būti standartizuoti.
- Atsargiai reikia elgtis su kategoriniais duomenimis.
- Nelabai tinka išankstiniam požymių atrinkimui prognozavimui.

# Reikalaujantis daug laiko resursų

Sudėtingas skaičiavimas: t-SNE reikalauja daug skaičiavimų, nes apskaičiuojamos kiekvieno duomenų taško porinės sąlyginės tikimybės ir stengiamasi sumažinti tikimybių skirtumo sumą aukštesniose ir mažesnioje dimensijoje. Dėl to jis ypač lėtas, skaičiavimo požiūriu gana sunkus ir reikalaujantis daug išteklių, kai taikomas duomenų rinkiniams, kuriuos sudaro daugiau kaip 10 000 stebėjimų. Kartais yra siūloma visų pirmą pritaikyti PCA, kad jos sumažintų kažkiek požymių ir tada taikyti t - SNE.

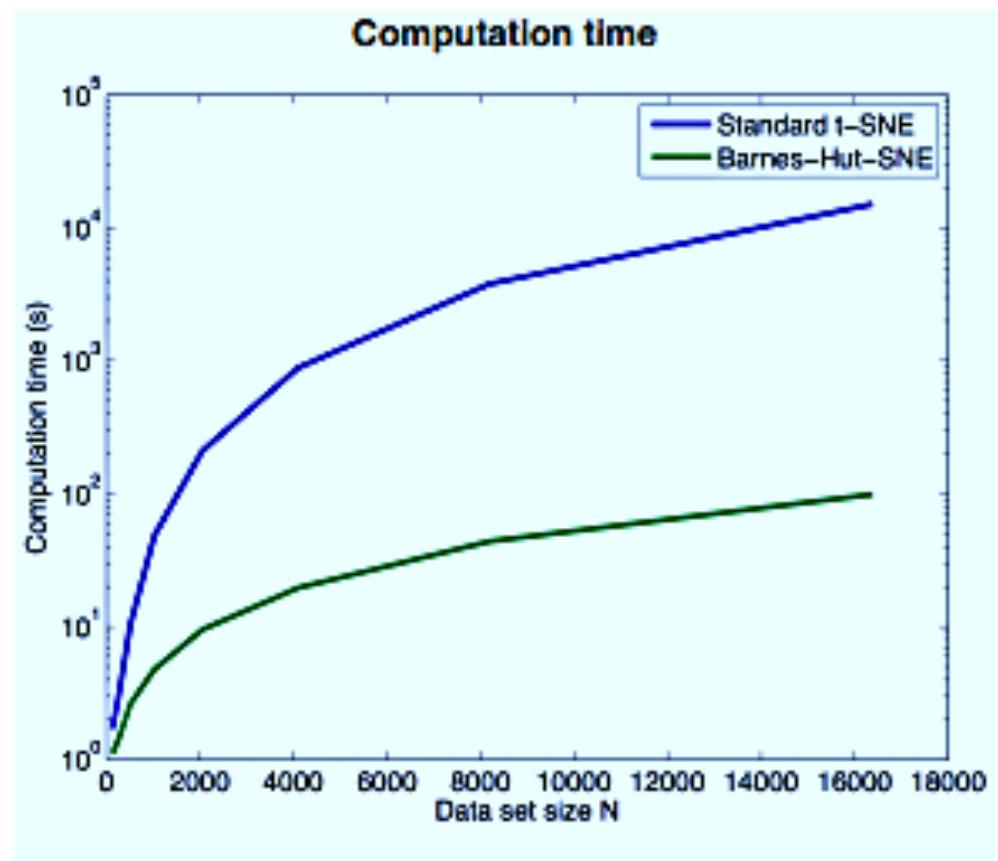
# Požymiu atrinkimas prognozavimui

Kai t - SNE modelį mokome, jis sutelkia dėmesį tik į konkrečias to duomenų rinkinio stebėjimų detales. Jis neišmoksta bendros funkcijos, kurią būtų galima veiksmingai taikyti nematytiems duomenims, kai, pavyzdžiui, gaunami nauji stebėjimai, pagal kuriuos norima atlikti prognozes

---

# MODIFIKACIJOS

# Barneso - Hut'o - SNE



van der Maaten, Laurens. (2013). Barnes-hut-sne. 1301.3342.

- Pagreitina algoritmo veikimą iš  $O(N^2)$  į  $O(N \times \log N)$ .
- Naudoja „vantage – point“ medžius tikimybių aproksimacijai. Tai yra subalansuotas medis, nes jis kaip slenkstį naudoja medianą, o ne vidurkį. Jei atstumas nuo vieno taško iki kito yra mažesnis už medianą, keliauja į kairijį lapą, o kitu atveju į dešinijį medžio lapą.
- Naudoja Barneso – Hut'o algoritmą gradiento aproksimacijai.

# Algoritmo modifikacijos – skirtinges atstumų metrikos

Modifikuotame t-sne sąlyginės porų panašumo tikimybės apskaičiuojamos remiantis ne Euklido metrika, o mahalanobio. Ši modifikacija tinkama gedinams / klaidoms aptikti, kuriame vienu metu atsižvelgiama į vietinę struktūrą ir skirtinges kintamųjų mastelius. Tinkama metrika pramoninių procesų duomenims.

Mahalanobio atstumas apskaičiuojamas pasitelkiant kovariacių matricą:

$$d_M(x, y) = \sqrt{(x - y)^T \times S^{-1} \times (x - y)},$$

*kur  $x$  ir  $y$  taškų koordinatės, o  $S^{-1}$  kovariacių matrica.*

# Algoritmo modifikacijos – skirtinės atstumų metrikos

Hammingo atstumas tarp dviejų vienodo ilgio eilučių - tai pozicijų, kuriose atitinkami simboliai skiriasi, skaičius. Kitaip tariant, jis parodo mažiausią pakeitimų skaičių, reikalingą vienai eilutei pakeisti kita, arba mažiausią klaidą, dėl kurių viena eilutė galėjo būti pakeista kita, skaičių. Bendresniame kontekste Hammingo atstumas yra viena iš keleto eilučių metrikų, skirtų dviejų sekų redagavimo atstumui matuoti.

The diagram shows two words, "woodman" and "woodland", aligned vertically. Vertical lines connect corresponding characters: 'w' to 'w', 'o' to 'o', 'o' to 'o', 'd' to 'd', 'm' to 'l', 'a' to 'a', and 'n' to 'n'. The character 'm' has a blue vertical line, while 'l' has a red vertical line, indicating they are different. The character 'n' has a green vertical line, indicating it is present in "woodland" but absent in "woodman".

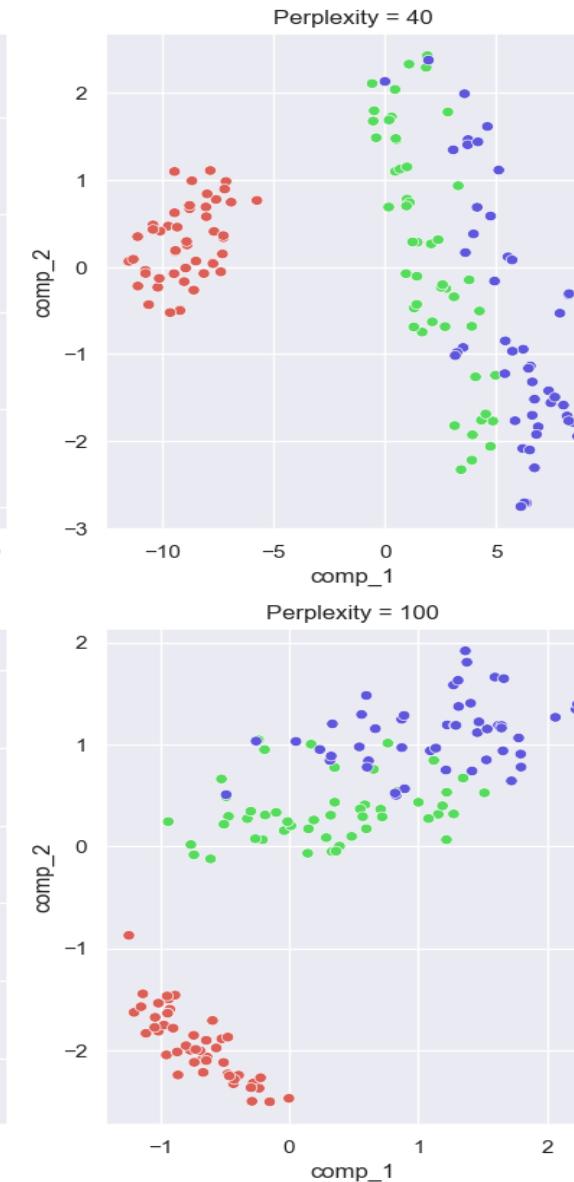
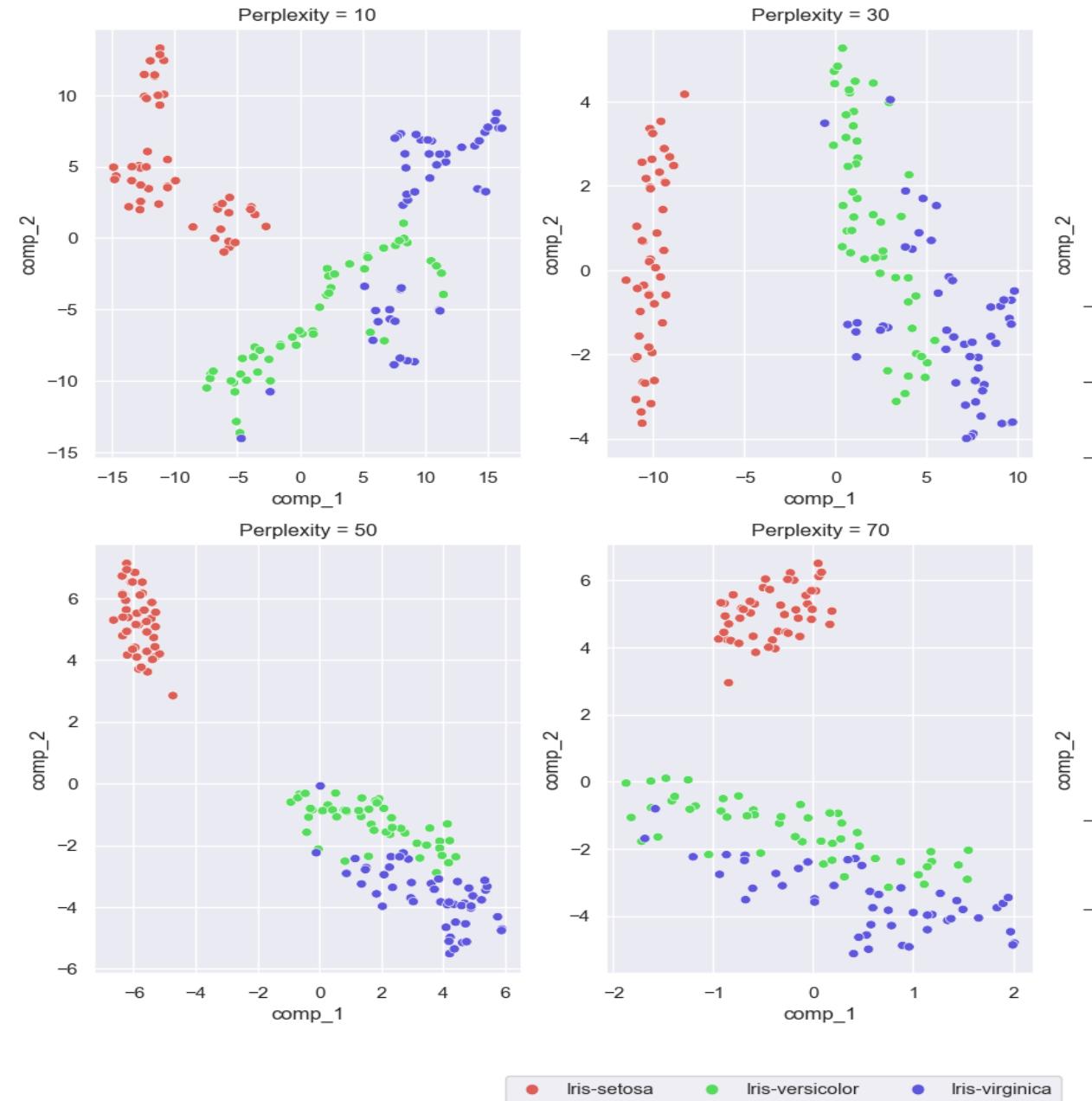
---

# PRITAIKYMAS DUOMENIMS

# Iris duomenų rinkinys

- 150 įrašų.
- Dimensija – 4.
- Duomenys suskirstyti į 3 klasses:
  - Iris setosa.
  - Iris versicolor.
  - Iris virginica.
- Kullback-Leiblerio divergencija
  - Po 250 iteracijų – 40,93.
  - Po 500 iteracijų – 0,08.

## Iris dataset projections with different perplexities



Vilniaus  
universitetas

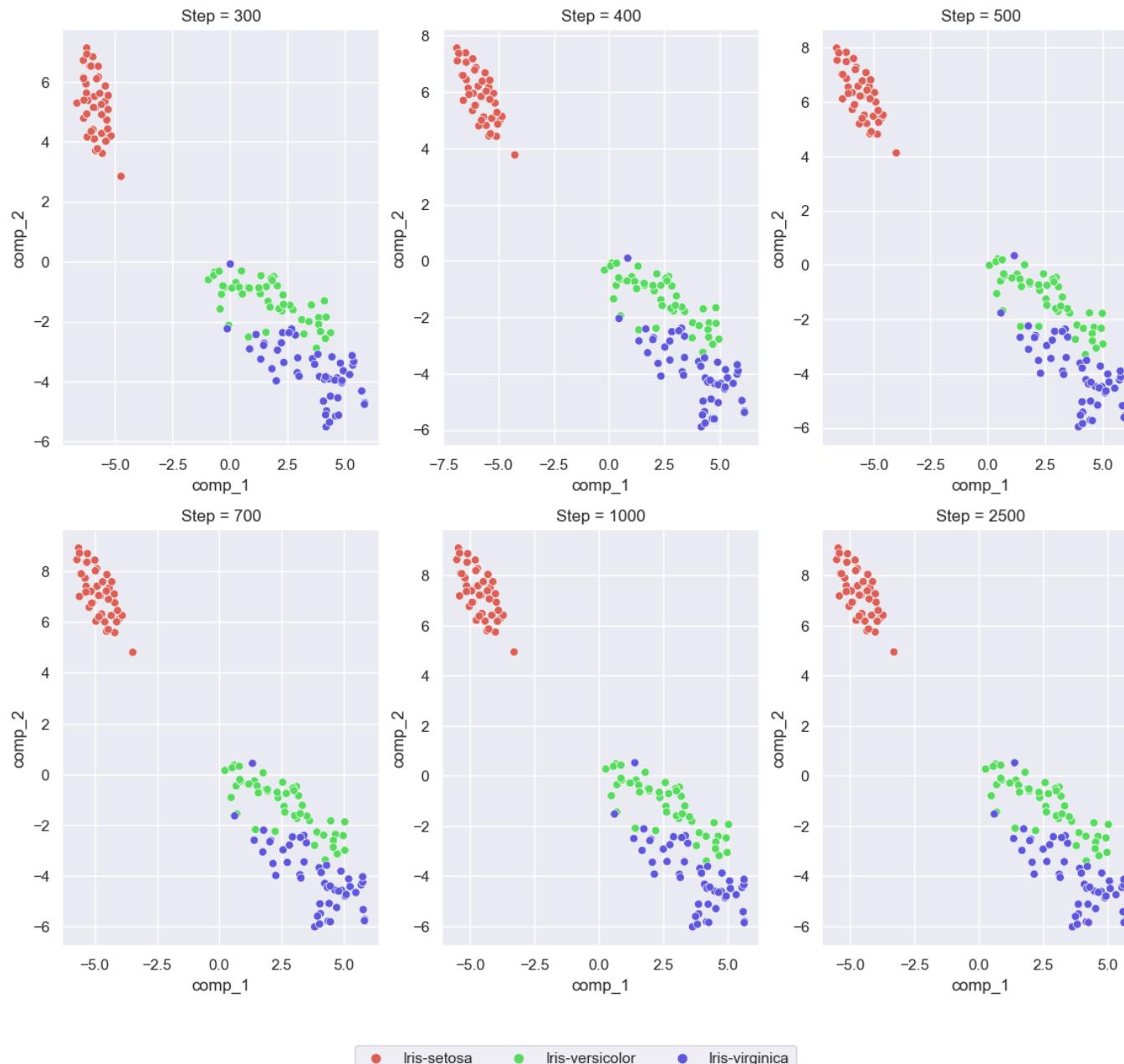
# *Iris* duomenų rinkinys

```
tsne = manifold.TSNE(  
    n_components = 2,           verbose=0, n_iter=300, random  
    state = 42)
```

## Keistas perpleksiškumo parametras.

Likusieji parametrai palikti su reikšmėmis pagal nutylėjimą.

Geriausias rezultatas gautas,  
kai perpleksišumas lygus 50.



# *Iris* duomenų rinkinys

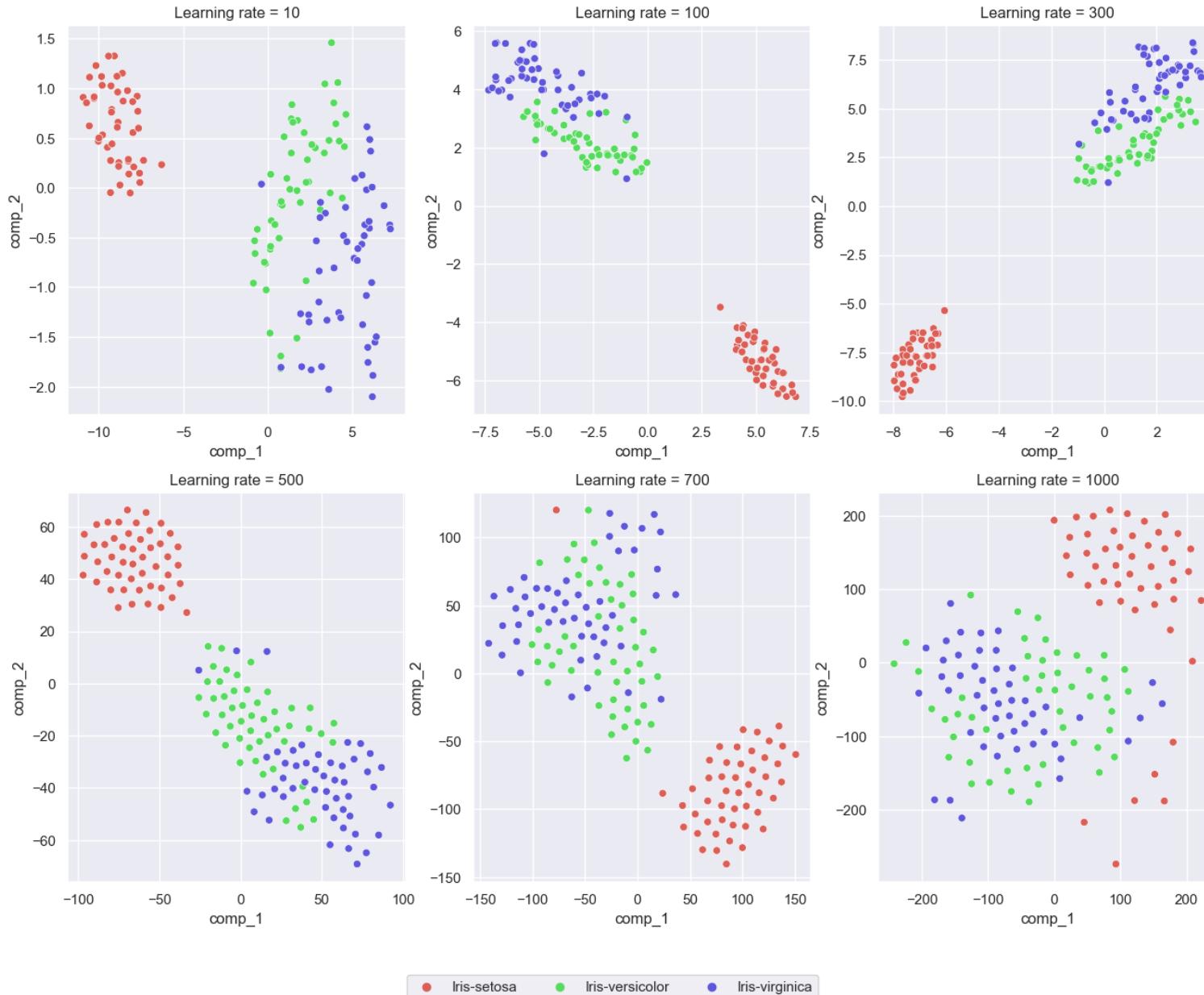
```
tsne = manifold.TSNE(n_componen
ts= 2, verbose=0, perplexity=50
, random_state = 42)
```

Keistas iteracijų skaičiaus parametras.

Fiksotas perpleksiškumo parametras lygus 50, kiti parametrai pagal nutylėjimą.

Rezultatai pasiekus 500 iteracijų nebesikeičia, todėl jį fiksuojame.

## Iris dataset projections with different learning rates



# Iris duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, verbose=0, perplexity = 50, n_iter=500, learning_rate = 1000, random_state = 42)
```

Fiksuoji perpleksiškumo ir iteracijų skaičiaus parametrai.

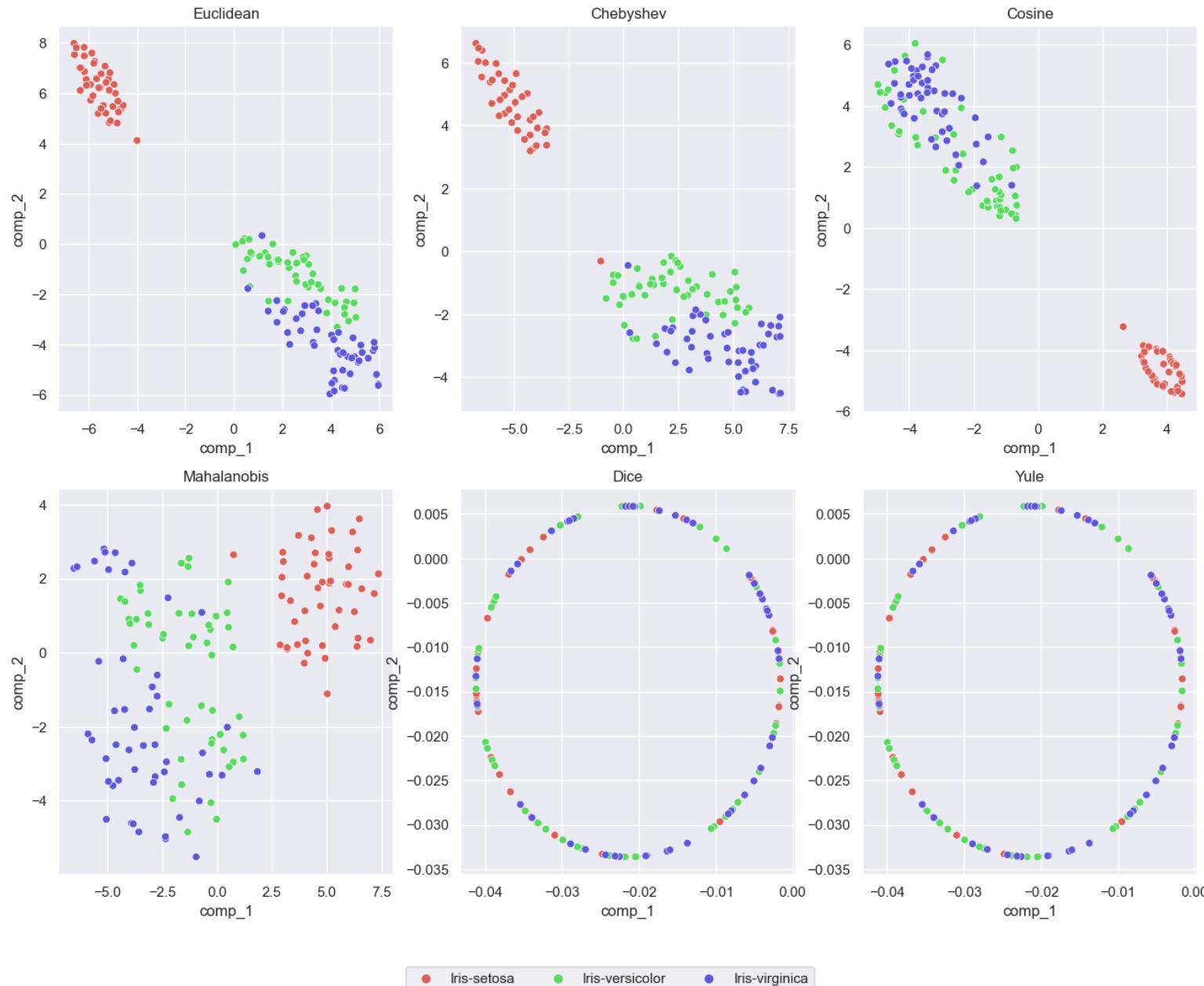
Keistas mokymo greičio parametras.

Geriausias rezultatas gautas, kai mokymo greitis lygus 100.

## Iris dataset projections with different metrics

# *Iris* duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, verbose=0, perplexity = 50, n_iter=500, random_state = 42)
```



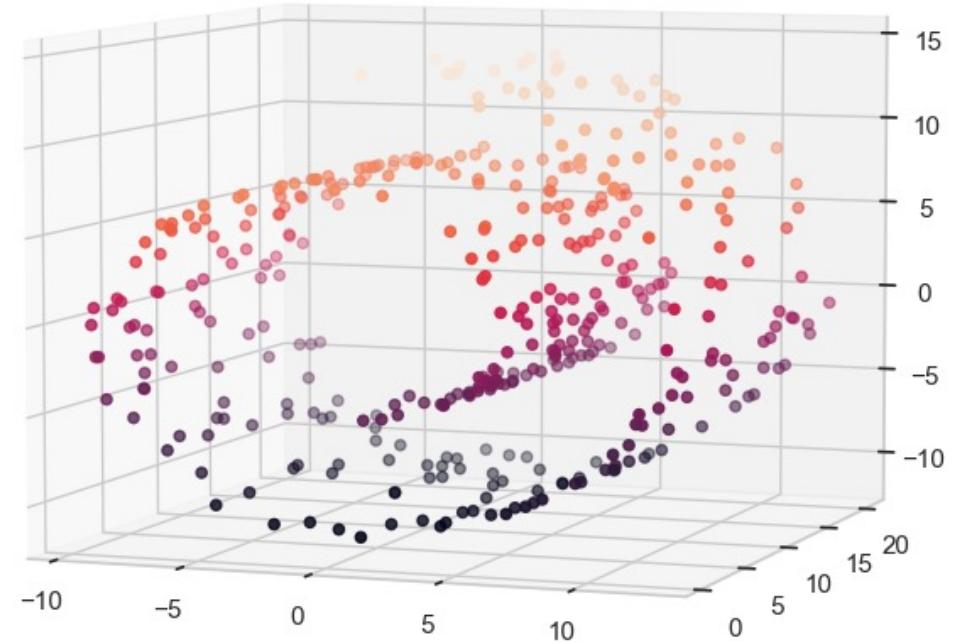
Fiksuoti perpleksiškumo ir iteracijų skaičiaus parametrai. Mokymo greičio nefiksavome, kad būtų taikomas „auto“ pagal nutylėjimą.

Keistas metrikos parametras.

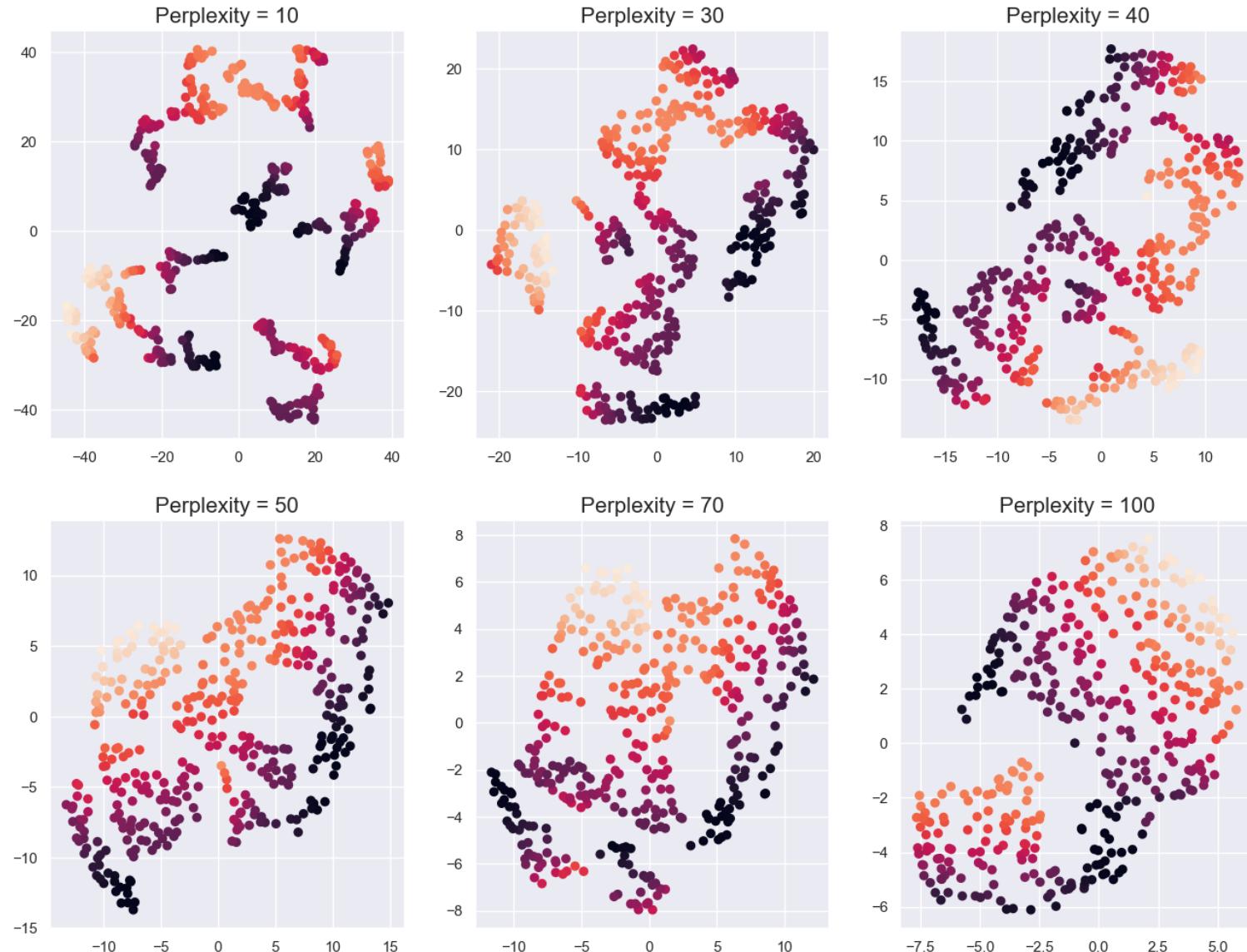
Geriausias rezultatas gautas su Euklidine metrika.

# *Swiss\_roll* duomenų rinkinys

- 400 įrašų.
- Dimensija – 3.
- Duomenys atitinka x, y ir z koordinates.
- Kullback - Leiblerio divergencija – 0,04.



Swiss\_roll dataset projections with different perplexities



# *Swiss\_roll* duomenų rinkinys

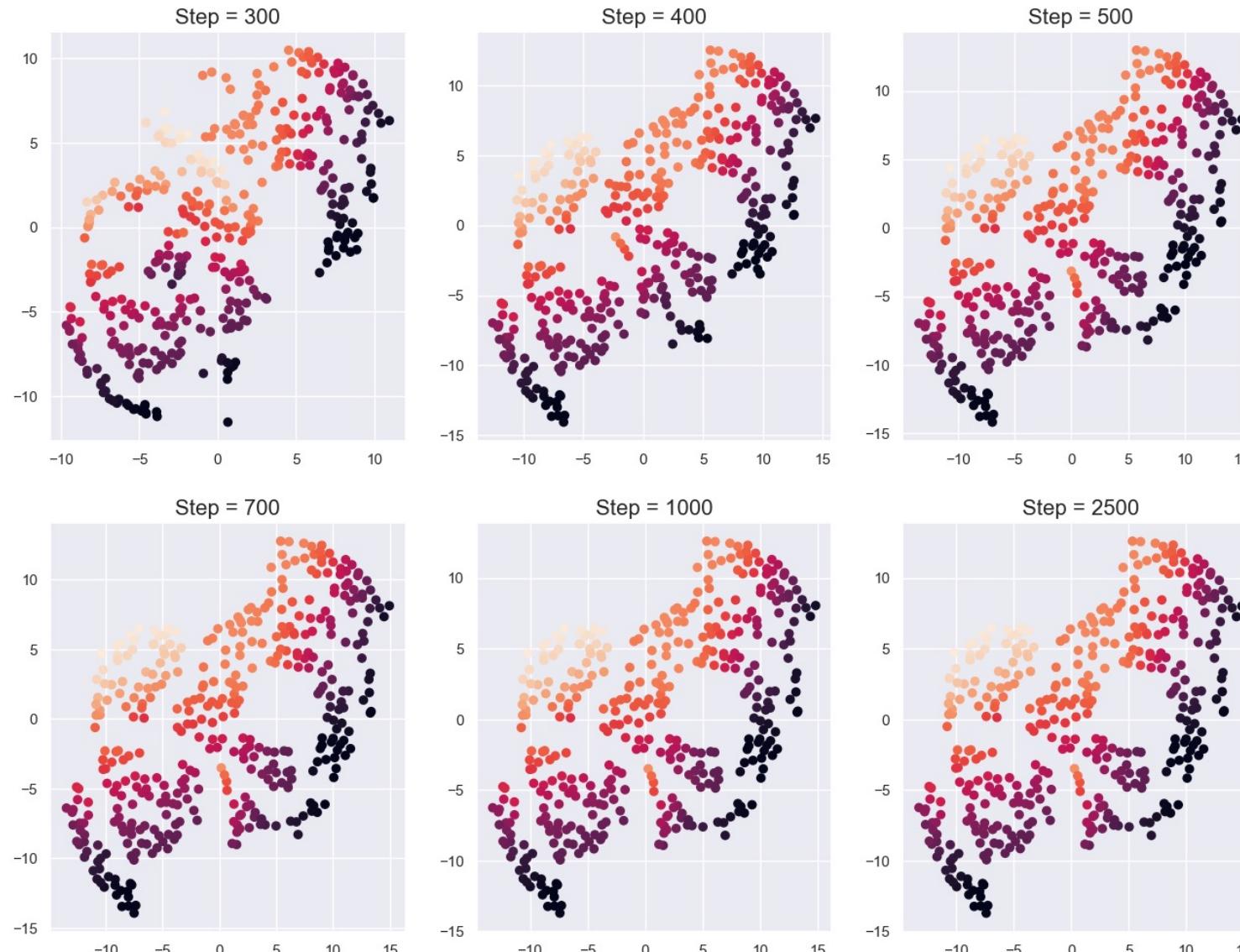
```
tsne = manifold.TSNE(n_components = 2, random_state = 42)
```

Keistas perpleksiškumo parametras.

Visi kiti parametrai palikti pagal nutylėjimą.

Geriausias rezultatas gautas su perpleksiškumu lygiu 50.

Swiss\_roll dataset projections with different iteration numbers



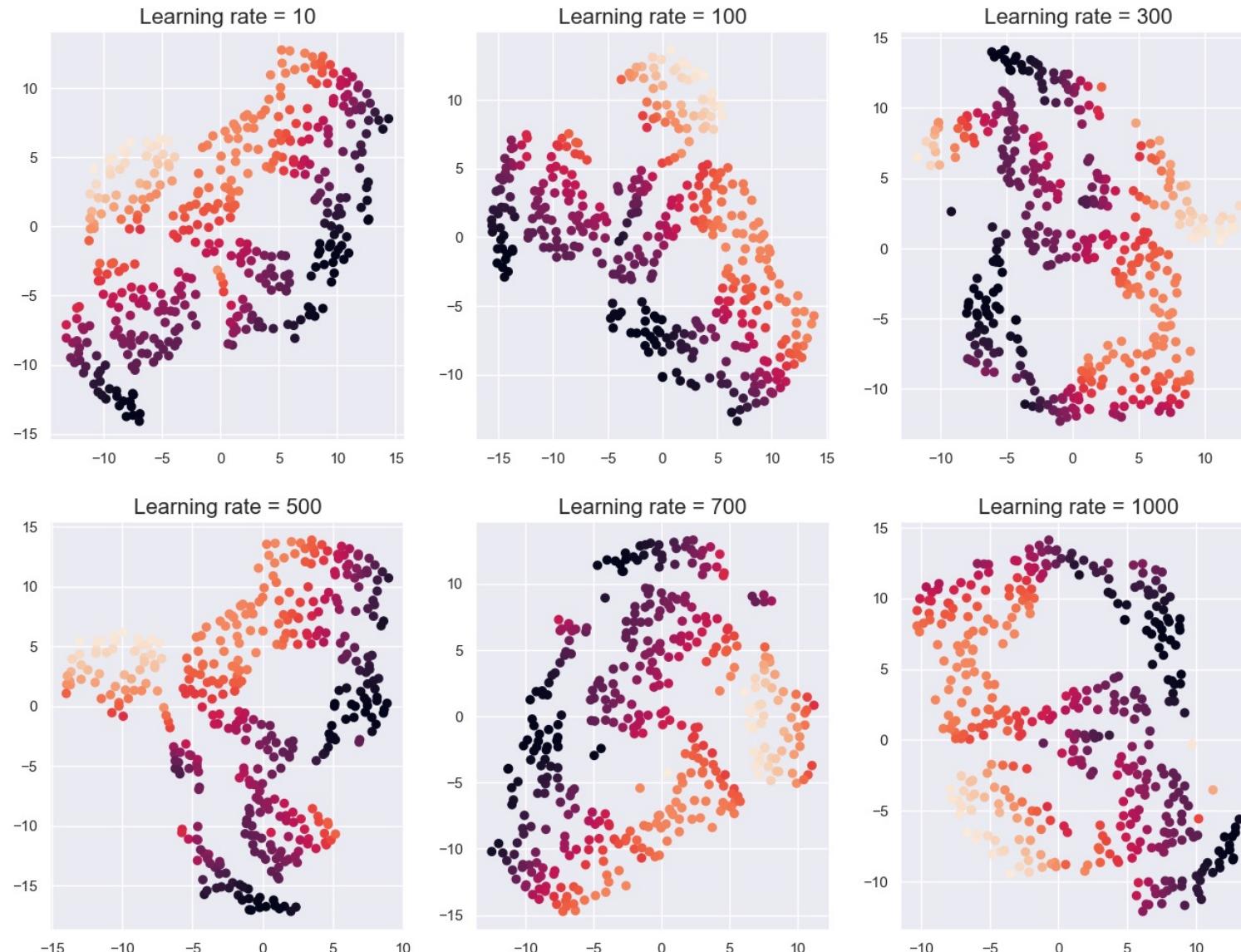
# Swiss\_roll duomenų rinkinys

```
tsne = manifold.TSNE(n_componen  
ts = 2, perplexity=50, random_st  
ate = 42)
```

Keistas iteracijų skaičius su fiksuotu perpleksiškumo parametru lygiu 50.

Rezultatai žymiai nesikeičia ir pasiekus 500 iteracijų nebesikeičia, todėl jį fiksuojame.

Swiss\_roll dataset projections with different learning rates



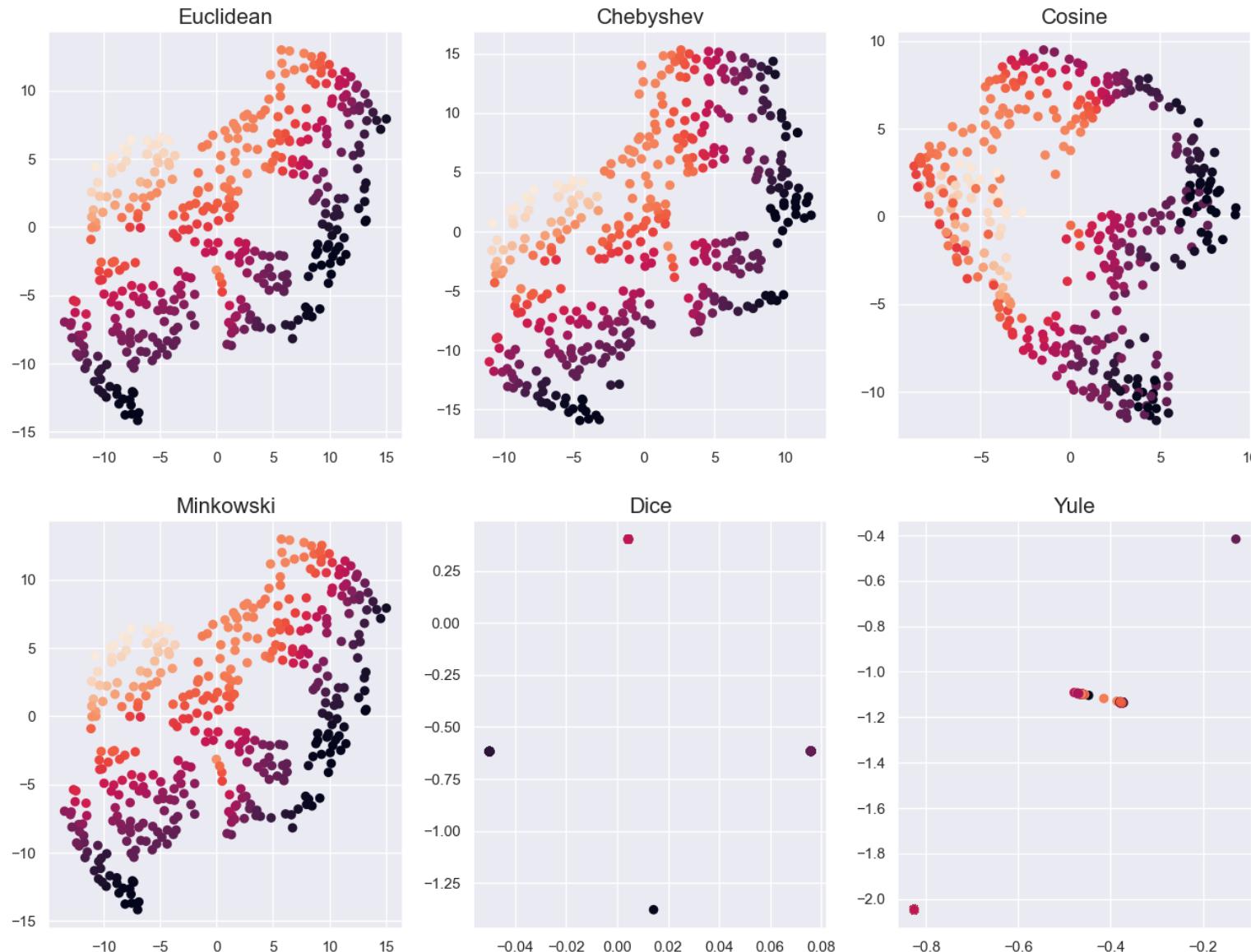
# Swiss\_roll duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, perplexity=50, n_iter = 500, random_state = 42)
```

Keistas mokymo greitis su fiksuotais perpleksiškumo (50) ir iteracijų skaičiaus (500) parametrais.

Geriausias rezultatas gautas, kai mokymo greitis lygus 100.

Swiss\_roll dataset projections with different metrics



# Swiss\_roll duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, perplexity=50, n_iter =500, random_state = 42)
```

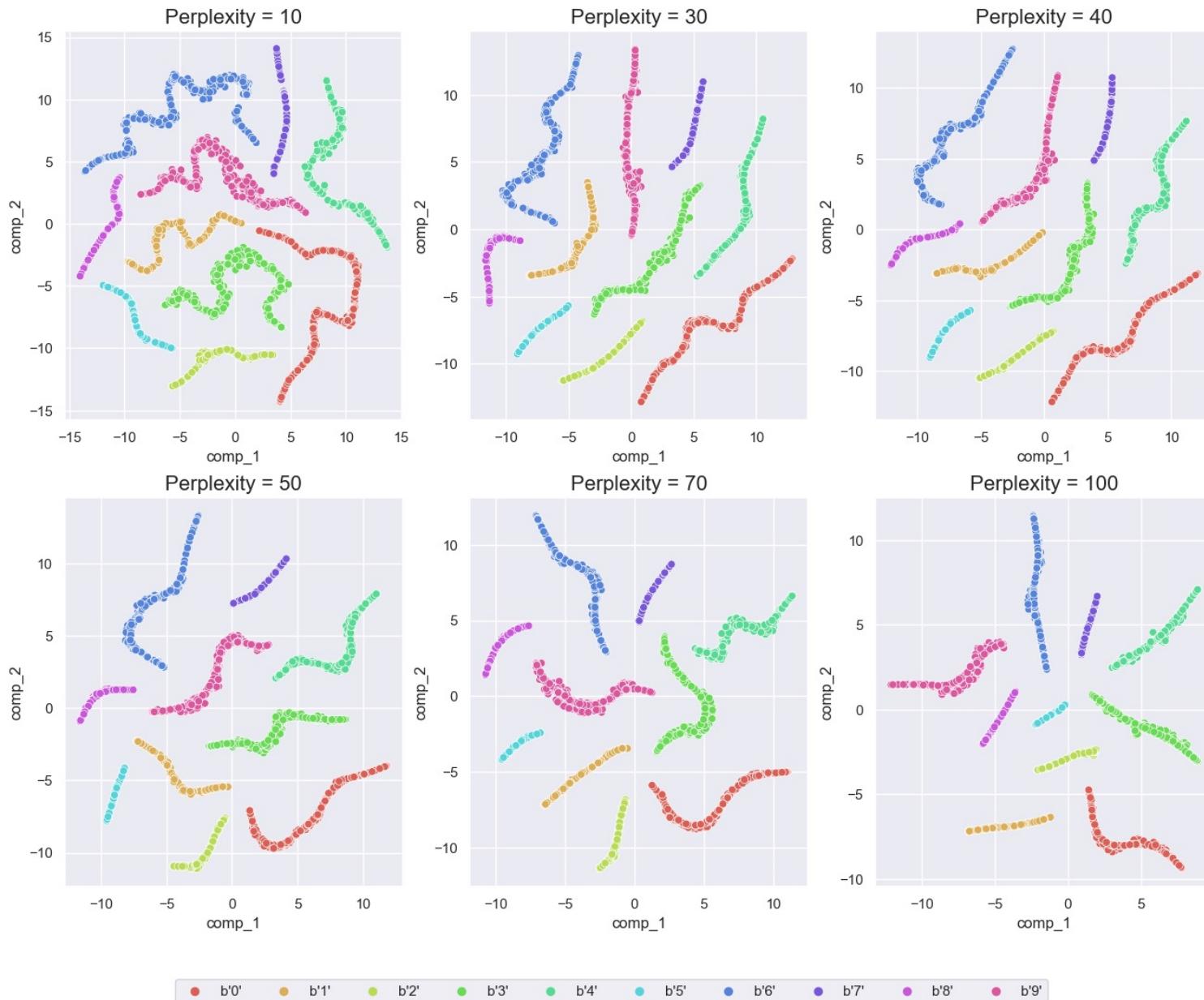
Keičiamos metrikos parametras, kai fiksotas 50 perpleksiškumas ir 500 iteracijų.

Euklidinė ir Minkovskio metrikos labai panašios ir geriausiai vaizduoja duomenis.

# *Ellipsoid* duomenų rinkinys

- 3140 įrašų.
- Dimensija – 50.
- Duomenys suskirstyti į 10 klasių.
- Kullback - Leiblerio divergencija:
  - Po 250 iteracijų – 52,15.
  - Po 300 iteracijų – 1,2.

Ellipsoid dataset projections with different perplexities



Vilnius  
universitetas

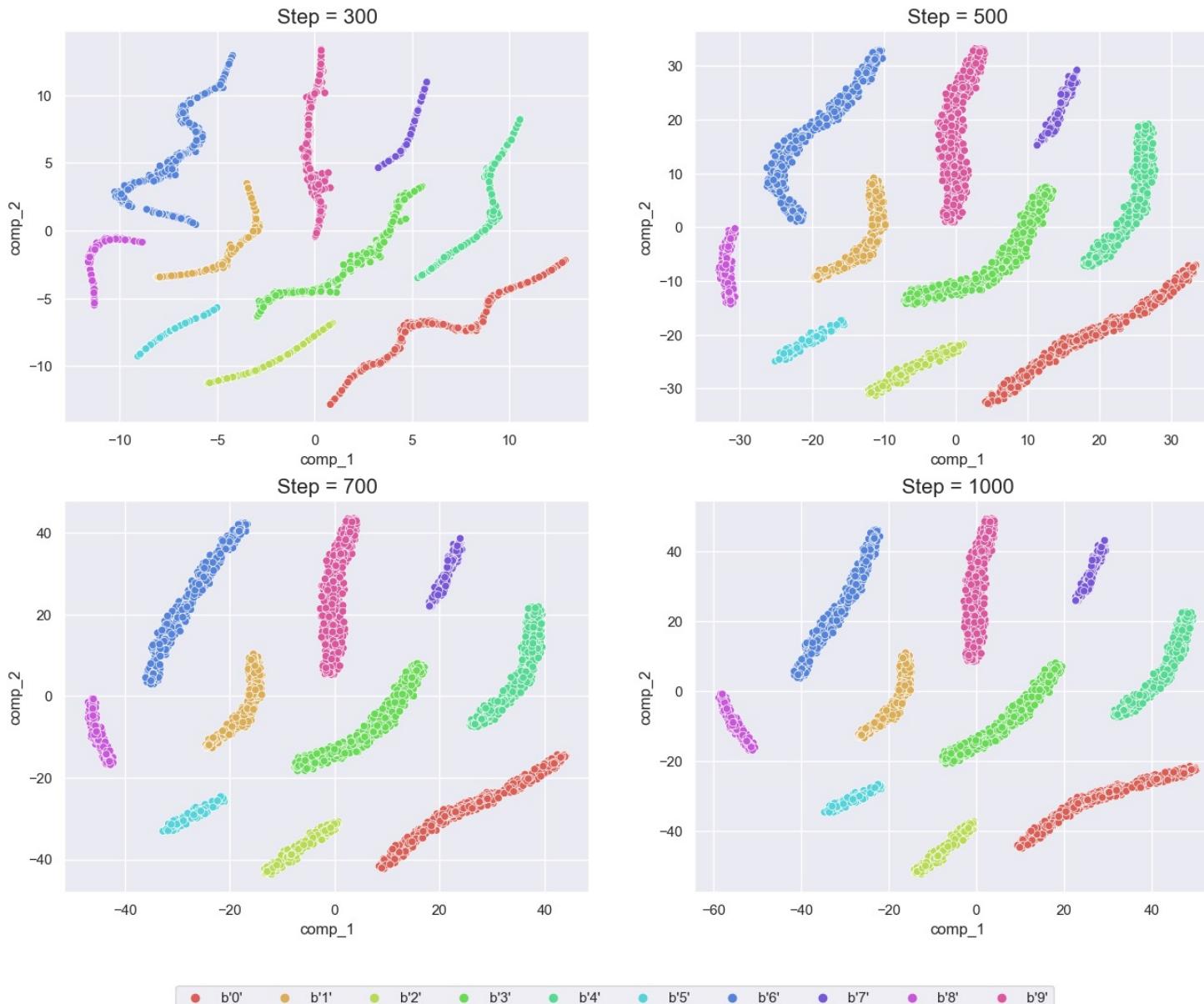
# Ellipsoid duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, verbose=0, n_iter=300, random_state = 42)
```

Keistas perpleksišumo parametras su fiksuotu 300 iteracijų skaičiumi.

Geriausias rezultatas gautas su perpleksišumo parametru lygiu 30.

## Ellipsoid dataset projections with different iteration numbers



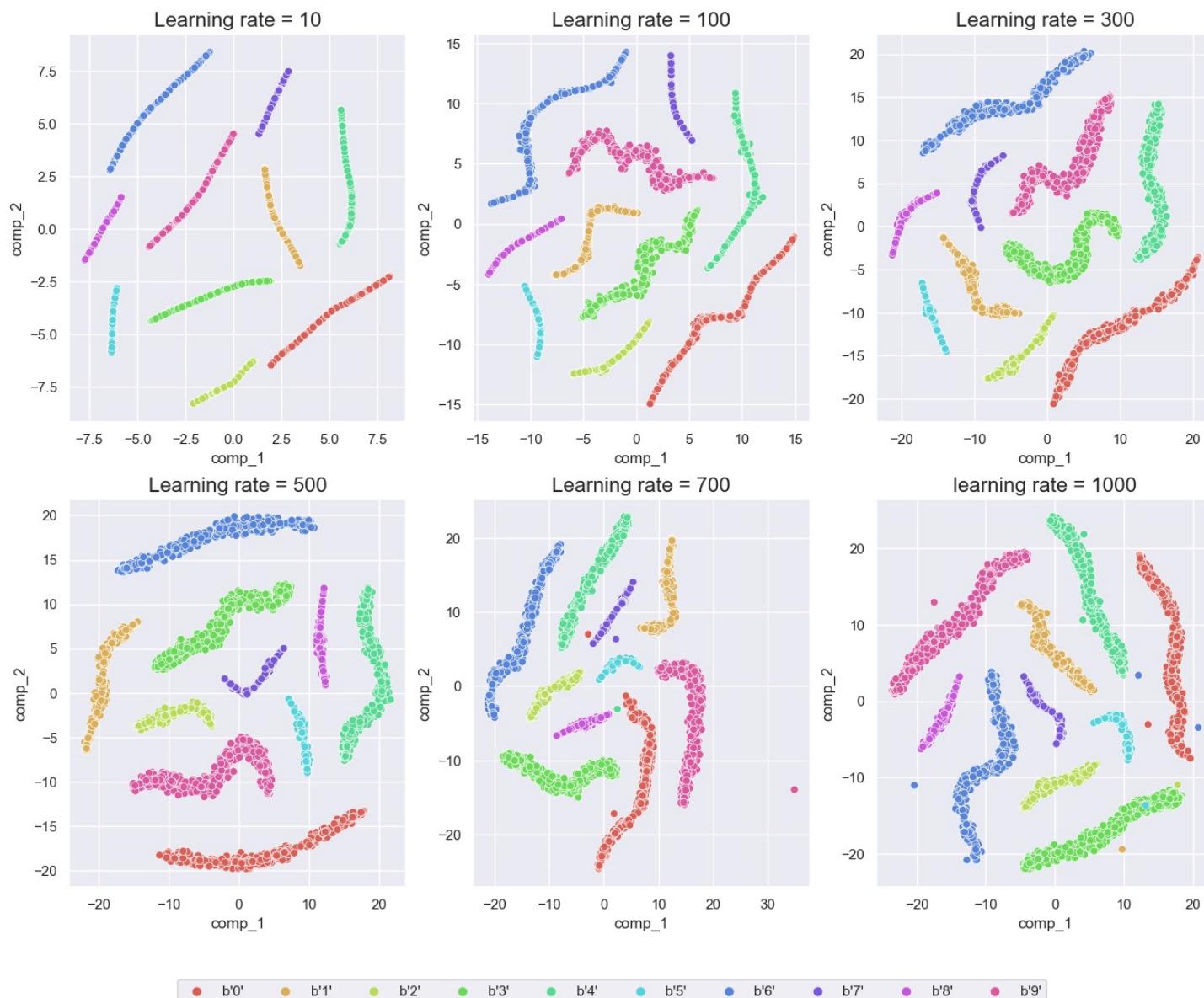
# Ellipsoid duomenų rinkinys

```
tsne = manifold.TSNE(n_components = 2, verbose=0, perplexity=30, random_state = 42)
```

Keičiamas iteracijų skaičius sufiksuoju 30 perpleksiškumu.

Geriausias rezultatas gautas, kai iteracijų skaičius lygus 300.

# Ellipsoid duomenų rinkinys

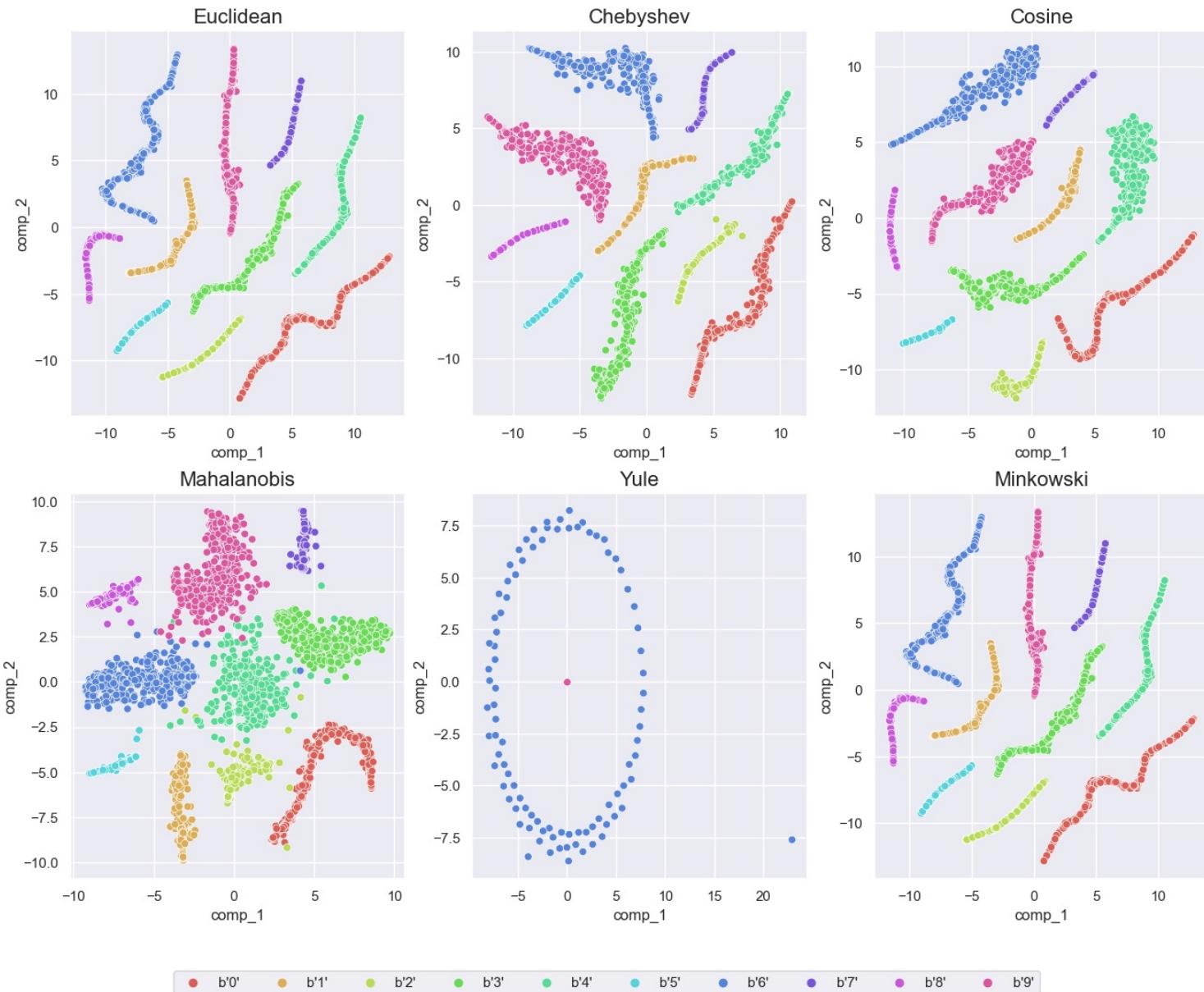


```
tsne = manifold.TSNE(n_components=2, verbose=0, perplexity=30, n_iter=300, random_state=42)
```

Keičiamas mokymo greitis, kai fiksuotas išlieka fiksuotas 30 perpleksiškumo parametras bei 300 iteracijų.

Geriausias rezultatas gautas, kai mokymo greitis lygus 100.

# *Ellipsoid* duomenų rinkinys



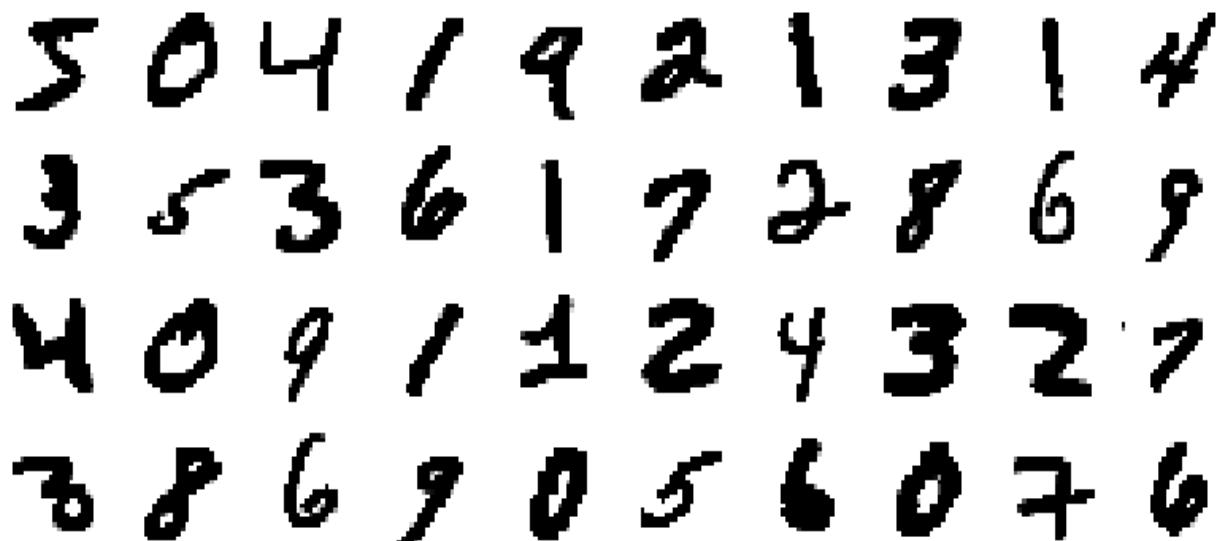
```
tsne = manifold.TSNE(n_components = 2, verbose=0, perplexity=30, n_iter=300, random_state = 42)
```

Keičiame metrikas, kai fiksuotas 30 perpleksiškumas ir 300 iteracijų.

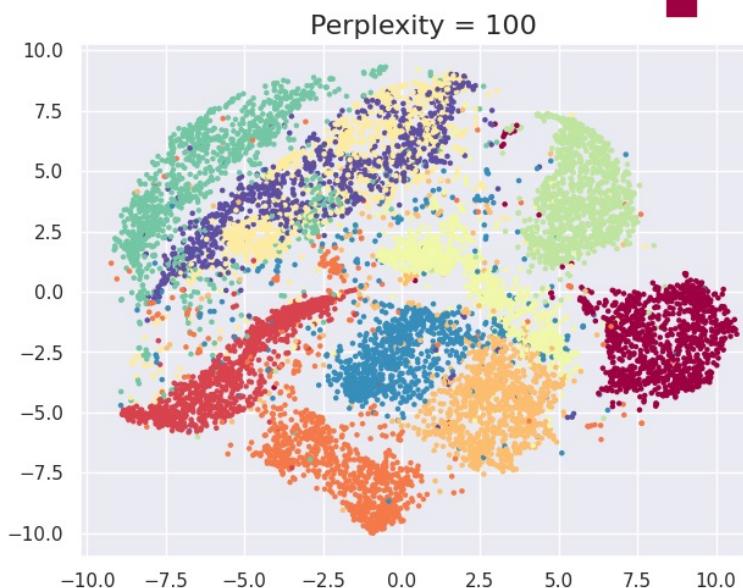
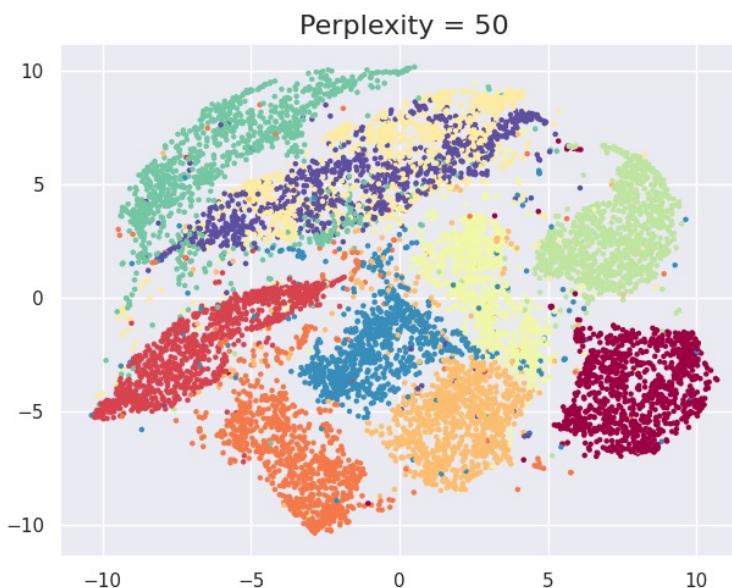
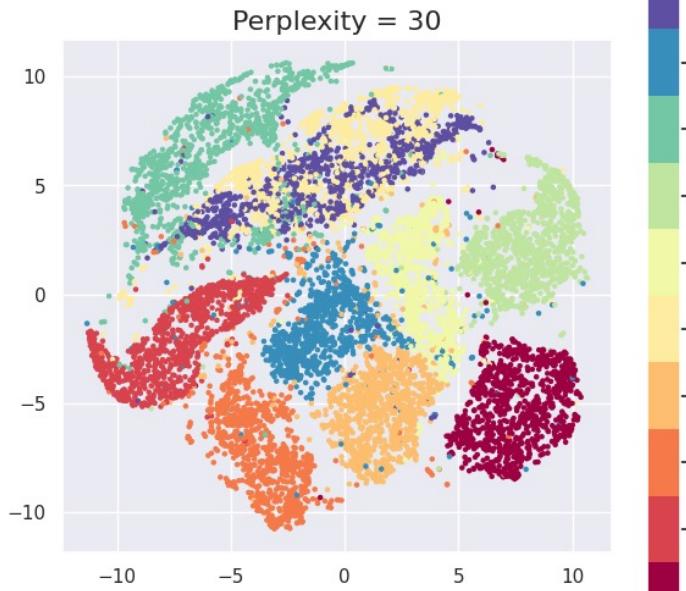
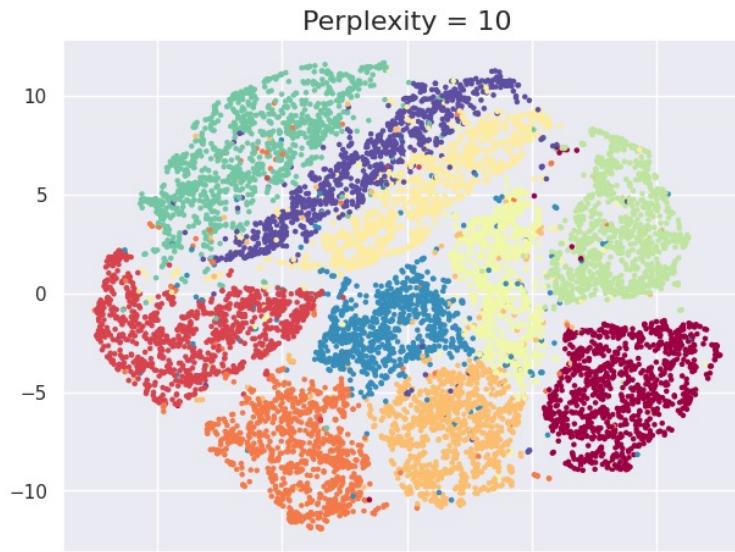
Geriausias rezultatas gautas su Euklidine ir Minkovskio metrika.

# Mnist duomenų rinkinys

- 60000 įrašų.
- Dimensija – 784.
- Duomenys suskirstyti į 10 klasių, kurios atitinka skaitmenis nuo 0 iki 9.
- Kullback - Leiblerio divergencija:
  - Po 250 iteracijų – 86,6.
  - Po 500 iteracijų – 2,03.



## MNIST dataset projections with different perplexities



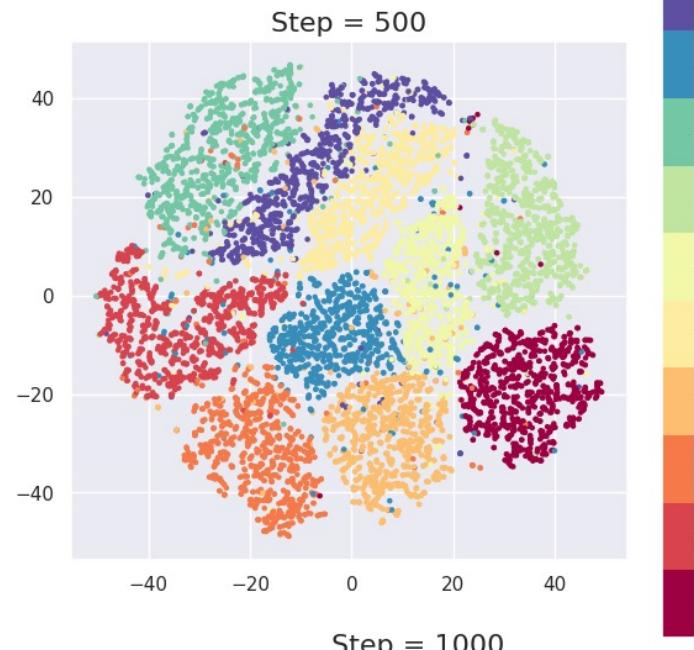
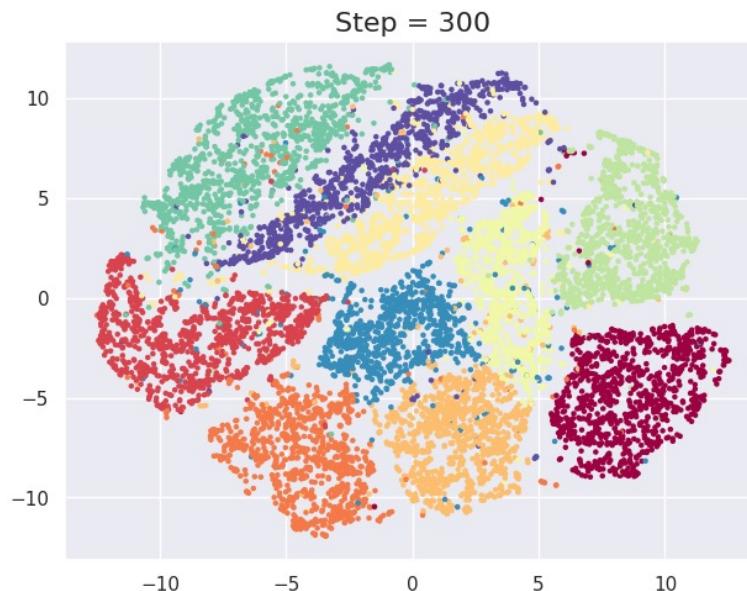
# Mnist duomenų rinkinys

```
tsne = TSNE(random_state=42, n_components=2, verbose=0, n_iter=300).fit_transform(x_subset)
```

Keistas perpleksiškumo parametras su fiksuotu 300 iteracijų skaičiumi.

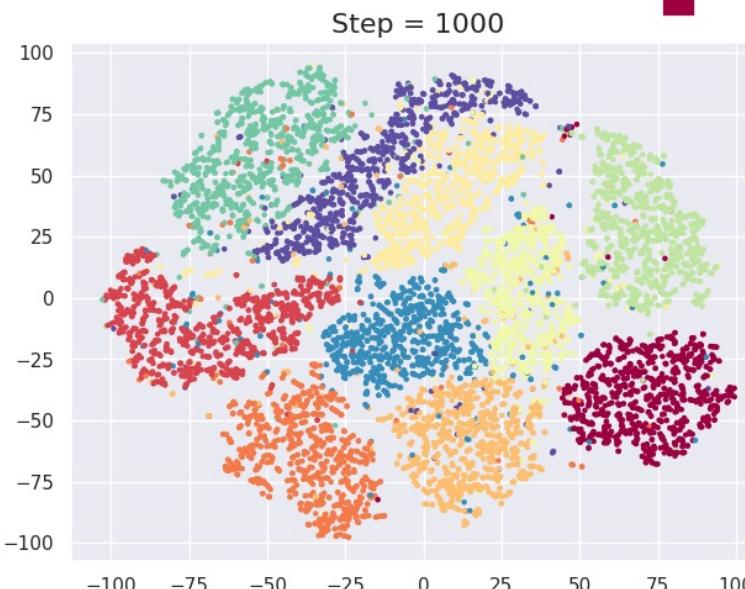
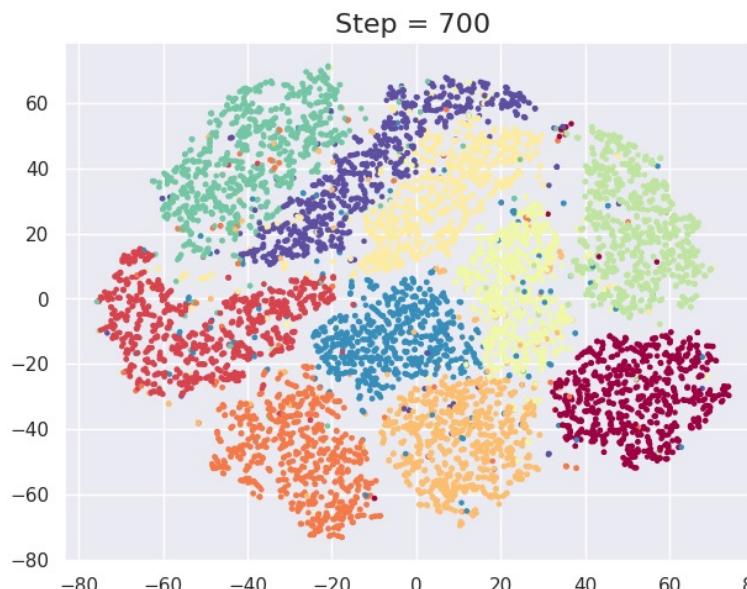
Geriausias rezultatas gautas su perpleksiškumo parametru lygiu 10.

## MNIST dataset projections with different iteration numbers



# Mnist duomenų rinkinys

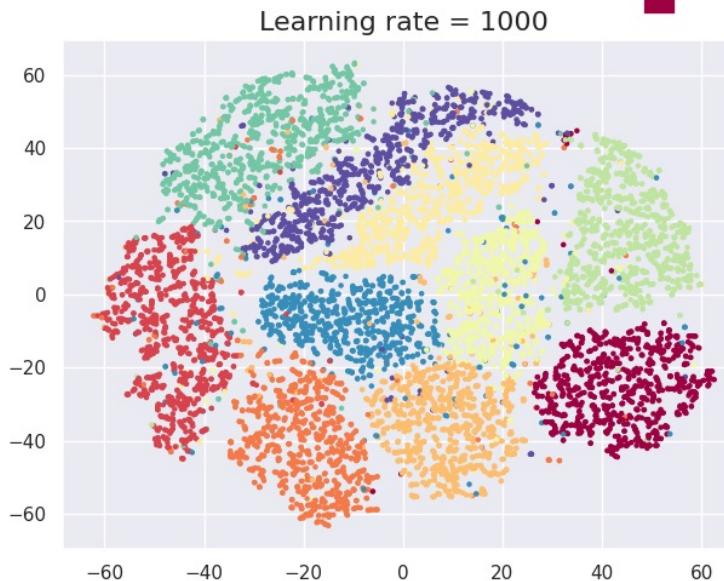
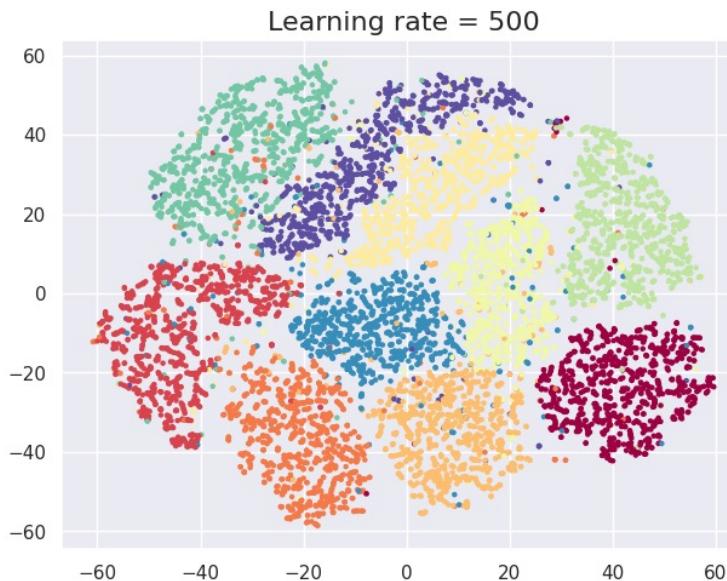
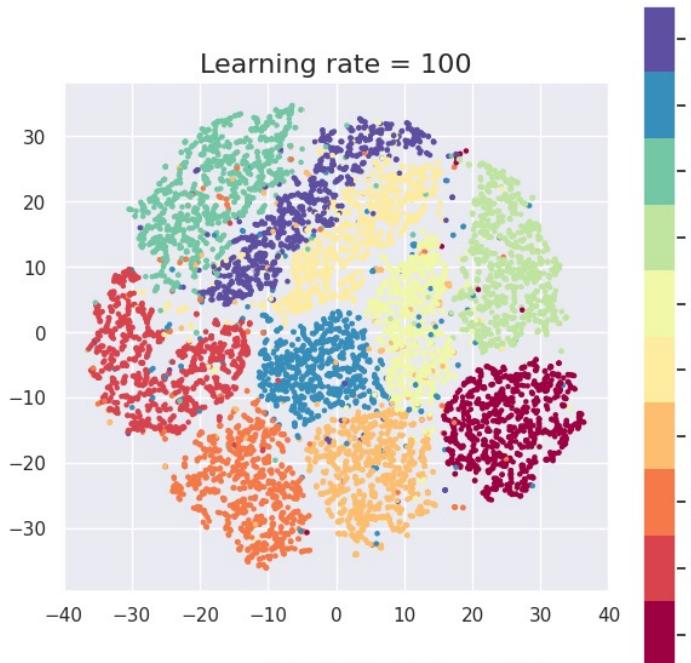
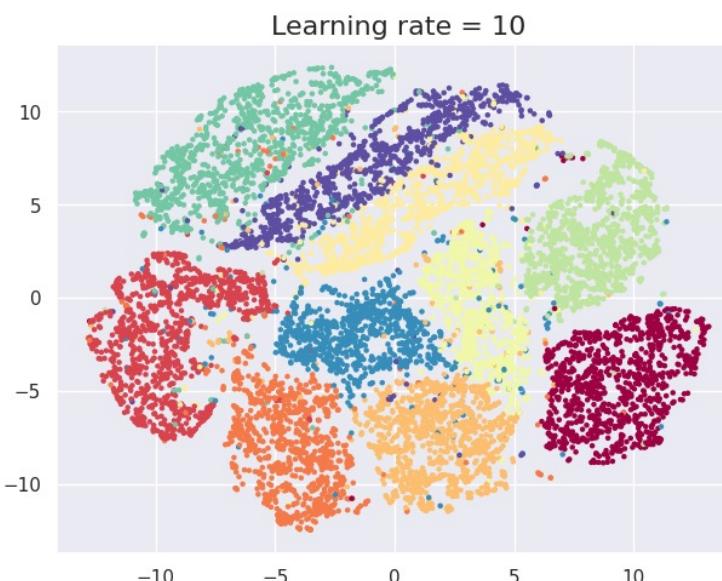
```
tsne10300 = TSNE(random_state = 42, n_components=2, verbose=0, perplexity=10).fit_transform(x_subset)
```



Keičiamas iteracijų skaičius sufiksuoju 10 perpleksiškumu.

Geriausias rezultatas gautas, kai iteracijų skaičius lygus 500.

## MNIST dataset projections with different learning rates



# Mnist duomenų rinkinys

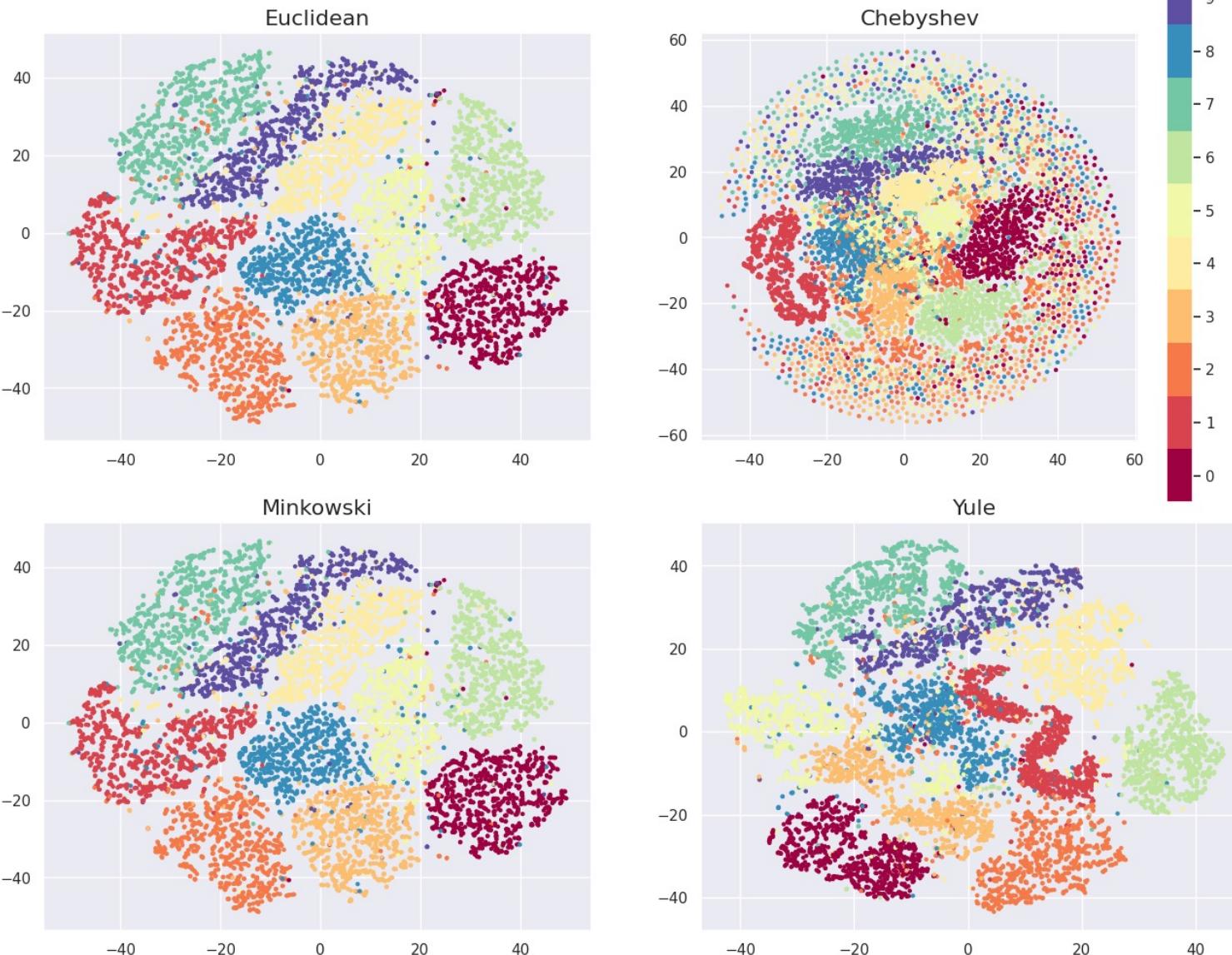
```
tsne = TSNE(random_state = 42, n_components=2, verbose=0, perplexity=10, n_iter=500).fit_transform(x_subset)
```

Keičiamas mokymo greitis, kai fiksuotas išlieka fiksuotas 10 perpleksiškumo parametras bei 500 iteracijų.

Geriausias rezultatas gautas, kai mokymo greitis lygus 500.

# Mnist duomenų rinkinys

```
tsne = TSNE(random_state =42,  
, n_components=2,verbose=0,  
perplexity=10,n_iter=500).  
fit_transform(x_subset)
```



Keičiame metrikas, kai fiksotas 10 perpleksišumas ir 500 iteracijų.

Geriausias rezultatas gautas su Euklidine ir Minkovskio metrikomis.

# Literatūra ir šaltiniai

- I. Wattenberg, et al., "How to Use t-SNE Effectively", Distill, 2016. <http://doi.org/10.23915/distill.00002>
- II. Y. Verma, „How to use t – SNE for dimensionality reduction?“, Mystery Vault, 2022. <https://analyticsindiamag.com/how-to-use-t-sne-for-dimensionality-reduction/#:~:text=t-SNE%20is%20a%20technique,data%20is%20very%20high%20dimensional>
- III. N. Shrivastav, „PCA vs t – SNE: which one should you use for visualization“, Analytics Vidhya, 2019. [https://medium.com/analytics-vidhya/pca-vs-t-sne-17bcd882bf3d#:~:text=t%2DSNE\(T%2D%20distributed%20Stochastic%20Neighbor%20Embedding\)%3A&text=One%20of%20the%20most%20major,large%20pairwise%20distance%20maximize%20variance.&text=It%20takes%20a%20set%20of,it%20into%20low%20dimensional%20data](https://medium.com/analytics-vidhya/pca-vs-t-sne-17bcd882bf3d#:~:text=t%2DSNE(T%2D%20distributed%20Stochastic%20Neighbor%20Embedding)%3A&text=One%20of%20the%20most%20major,large%20pairwise%20distance%20maximize%20variance.&text=It%20takes%20a%20set%20of,it%20into%20low%20dimensional%20data)
- IV. M. Lee, „What is tSNE and when should I use it?“, 2021. <https://sonraianalytics.com/what-is-tsne/#:~:text=T-distributed%20Stochastic%20Neighbourhood%20Embedding,in%202%20or%203%20dimensions>
- V. A. Kumar Pal, „Dimension Reduction – t – SNE“, 2018. <https://blog.paperspace.com/dimension-reduction-with-t-sne/>
- VI. K. Erdem, „t-SNE clearly explained“, 2020. <https://towardsdatascience.com/t-sne-clearly-explained-d84c537f53a#:~:text=t-SNE%20is%20mostly%20used,when%20dealing%20with%20CNN%20networks>
- VII. N. Kumar, „Advantages and Disadvantages of t – SNE over PCA (PCA vs t – SNE). <http://theprofessionalspoint.blogspot.com/2019/03/advantages-and-disadvantages-of-t-sne.html>

# Literatūra ir šaltiniai

- VIII. C. Ellis, „When to use t – sne“. <https://crunchingthedata.com/when-to-use-t-sne/>
- IX. H. Agarwal, „t – SNE (t – Distributed Stochastic Neighbor Embedding) Algorithm“, enjoy algorithms, <https://www.enjoyalgorithms.com/blog/tsne-algorithm-in-ml>
- X. „Comprehensive Guide on t – SNE algorithm with implementation in R & Python“, 2022. <https://www.analyticsvidhya.com/blog/2017/01/t-sne-implementation-r-python/>
- XI. C. Rossant, „An illustrated introduction to the t – SNE algorithm“, 2015. <https://www.oreilly.com/content/an-illustrated-introduction-to-the-t-sne-algorithm/>
- XII. D. Liu, T. Guo and M. Chen, "Fault Detection Based on Modified t-SNE," 2019 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS), Xiamen, China, 2019, pp. 269-273, doi: 10.1109/SAFEPROCESS45799.2019.9213365. <https://ieeexplore.ieee.org/document/9213365>
- XIII. R. Wang, X. Zhang, „Capacity Preserving Mapping for High – dimensional Data visualization, 2021. <https://arxiv.org/pdf/1909.13322.pdf#:~:text=Simply%20put%2C%20the%20crowding%20issue,causes%20points%20to%20be%20crowded>
- XIV. Firuza, „Vantage Point Tree: does careful selection of vantage point make sense?“, 2018. <https://firuza.medium.com/vantage-point-tree-does-careful-selection-of-vantage-point-make-sense-c74bbd6d77e9>

*AČIŪ UŽ DĘMESŁ!*