Duomenų tyrybos praktinė užduotis

Darbą atliko: Vytautas Žilinas (Magistro 1 kurso studentas)

Įžanga

Šiame darbe nagrinėjamas Future-500-17.csv. Duomenų aibei nagrinėti naudojamas Python 3.8
Užduoties tikslas – išanalizuoti duotų duomenų aibę, atlikti pirminį duomenų apdorojimą: užpildyti praleistus duomenis, išskirti taškus atsiskyrėlius, pritaikyti kelis normavimo metodus, pateikti aprašomąsias duomenų statistikas. Atlikti tiriamos aibės vizualią analizę, naudojant taškines, stačiakampes diagramas, histogramas, dimensijos mažinimo algoritmus.

Uždaviniai:

1. Aprašyti užduoties tikslą ir uždavinius.
2. Trumpai aprašyti tiriamą duomenų aibę, kokie požymiai: skaitiniai, ranginiai ir pan.?
3. Pateikti atskirų požymių aprašomąsias statistikas lentelės pavidalu: min, max, 1, 3 kvartilės, vidurkis mediana, dispersija ir pan.
4. Pasirinktais metodais užpildyti praleistas reikšmes, mokėti argumentuoti, kokį metodą taikėte ir kodėl.
5. Nustatyti taškus atsiskyrėlius, pašalinti juos iš duomenų aibės, palyginti, kaip pasikeitė imties statistiniai duomenys.
6. Sunormuoti duomenų aibę naudojant du normavimo metodus: pagal vidurkį ir dispersiją, min - max.
7. Pateikti vizualią duomenų aibės analizę: taškiniai grafikai, dažnio diagramos, histogramos, stačiakampės diagramos. Po kiekvienu grafiku turi būti interpretacija, kokias išvadas gauname analizuojant grafikus. Kaip pajamos priklauso nuo pramonės šakos? Koks pelno pasiskirstymas pagal valstijas? Ir t.t.
8. Apskaičiuoti požymių koreliacijas, pateikti skaitinius įverčius lentelės pavidalu.
9. Duomenų aibę suformuoti paliekant tik skaitinius požymius ir Industry stulpelį. Vizualizuoti daugiamačius duomenis naudojant PCA ir MDS algoritmus.
10. Reikia pateikti atliekamos užduoties kodus.

Duomenų aibė

Imties dydis - Nagrinėjami duomenys susidaro iš 500 įmonių.

Imties duomenų savybės - Nagrinėjama duomenų aibė susidaro iš požymių: ID, Name, Industry, Inception, Employees, State, City, Revenue, Expenses, Profit, Growth.

Šios savybės skaidomi į šiuos tipus:

Nominalieji: Industry, Inception, State, City,

Ranginiai: ID

Kiekybiniai diskretieji: Revenue, Expenses, Profit, Employees

Tolydieji: Growth

Duomenų priešanalizė

Neapdorotus duomenis analizuojant su Python priedu Pandas

```
data = pd.read_csv("Future-500-17.csv")
print(data.describe(include='all'))
```

	ID	Name	Industry	Inception	Employees	State	City	Revenue	Expenses	Profit	Growth
count	500.000000	500	497	499.000000	495.000000	495	500	493	495	4.970000e+02	497
unique	NaN	500	7	NaN	NaN	42	297	493	495	NaN	32
freq	NaN	1	145	NaN	NaN	57	13	1	1	NaN	39
mean	250.500000	NaN	NaN	2010.174349	149.161616	NaN	NaN	NaN	NaN	6.534190e+06	NaN
std	144.481833	NaN	NaN	3.228211	398.474670	NaN	NaN	NaN	NaN	3.872034e+06	NaN
min	1.000000	NaN	NaN	1999.000000	1.000000	NaN	NaN	NaN	NaN	1.243400e+04	NaN

	ID	Name	Industry	Inception	Employees	State	City	Revenue	Expenses	Profit	Growth
25%	125.750000	NaN	NaN	2009.000000	27.500000	NaN	NaN	NaN	NaN	3.259485e+06	NaN
50%	250.500000	NaN	NaN	2011.000000	56.000000	NaN	NaN	NaN	NaN	6.512379e+06	NaN
75%	375.250000	NaN	NaN	2012.000000	126.000000	NaN	NaN	NaN	NaN	9.314149e+06	NaN
max	500.000000	NaN	NaN	2014.000000	7125.000000	NaN	NaN	NaN	NaN	1.962453e+07	NaN

Gauname tokius rezultatus iš kurių matome, jog trūksta duomenų visur išskyrus Name, Industry. Todėl duomenis turime apvalyti.

Praleistų reikšmių užpildymas

Rankomis užpildomi State duomenys, kadangi juos galima gauti pagal City stulpelį.

Duomenų išvalymas:

Kategoriniams duomenims uždedami tipai:

```
data['Industry'] = data['Industry'].astype('category')
data['Name'] = data['Name'].astype('category')
data['Inception'] = data['Inception'].astype('category')
data['State'] = data['State'].astype('category')
data['City'] = data['City'].astype('category')
```

Like duomenys užpildomi Python pagalba:

Expenses stulpeliui šalinami "Dollars" ir kableliai:

```
data['Expenses'] = data['Expenses'].str.replace("Dollars", "")
data['Expenses'] = data['Expenses'].str.replace(",", "")
data['Expenses'] = pd.to_numeric(data['Expenses'], errors='coerce', downcast='float')
```

Expenses stulpeliui šalinami "Dollars" ir kableliai:

```
data['Expenses'] = data['Expenses'].str.replace("Dollars", "")
data['Expenses'] = data['Expenses'].str.replace(",", "")
data['Expenses'] = pd.to_numeric(data['Expenses'], errors='coerce', downcast='float')
```

Revenue stulpeliui šalinami "$" ir kableliai:

```
data['Revenue'] = data['Revenue'].str.replace("$", "")
data['Revenue'] = data['Revenue'].str.replace(",", "")
data['Revenue'] = pd.to_numeric(data['Revenue'], errors='coerce', downcast='float')
```

Growth šalinamas procentų ženklas ir dalinama iš 100:

```
data['Growth'] = data['Growth'].str.replace("%", "")
data['Growth'] = pd.to_numeric(data['Growth'], errors='coerce', downcast='float') / 100
```

Profit ir Employees nustatomas skaitinis tipas:

```
data['Profit'] = pd.to_numeric(data['Profit'], errors='coerce', downcast='float')
data['Employees'] = pd.to_numeric(data['Employees'], errors='coerce', downcast='float')
```

Revenue ir Employees užpildomi pagal Industry stulpelio medianą:

```
data['Revenue'].fillna(data.groupby('Industry')['Revenue'].transform('median'), inplace=True)
data['Employees'].fillna(data.groupby('Industry')['Employees'].transform('median'), inplace=True)
```

Bandoma užpildyti Expenses ir Profit naudojant formulę (Expenses = Revenue - Profit):

```
data['Expenses'] = data['Expenses'].fillna(data['Revenue'] - data['Profit'])
data['Profit'] = data['Profit'].fillna(data['Revenue'] - data['Expenses'])
```

Nežinomiems Growth nustatomas 0:

```
data['Growth'] = data['Growth'].fillna(value=0)
```

Tas eilutes kurių Revenue, Expenses, Profit, Industry nepavyko išskaičiuoti yra šalinamos:

```
data = data[data['Revenue'].notna() & data['Expenses'].notna()
            & data['Profit'].notna() & data['Industry'].notna()]
```

Sutvarkius duomenys gaunami tokie rezultatai:

	Employees	**Revenue**	**Expenses**	**Profit**	**Growth**
count	495.000000	495.0	495.00	495.00	495.000000
mean	148.870712	10831591.0	4297555.50	6532033.00	0.143232
std	398.469299	3190166.5	2125169.75	3871154.25	0.069440
min	1.000000	1614585.0	-41678.00	12434.00	-0.030000
25%	28.000000	8696234.5	2755930.00	3284662.50	0.080000
50%	56.000000	10651148.0	4316632.00	6512379.00	0.150000
75%	125.500000	13096431.0	5814274.00	9293752.50	0.200000
max	7125.000000	21810052.0	9860686.00	19624534.00	0.300000

Taškų atsiskyrėlių arba išskirčių identifikavimas ir išmetimas

Buvo pasirinkta šalinti Employees, Revenue, Expenses, Profit, Growth ekstremalius atsiskyrėlius Buvo panaudotas toks pats kodas visiems šiems elementams, vaizduojamas su Revenue pavyzdžiu:

```
q_low = data["Revenue"].quantile(0.25)
q_hi = data["Revenue"].quantile(0.75)
q_dif = 3 * (q_hi - q_low)
q_dif = 3 * (q_hi - q_low)
df_filtered = data[(data["Revenue"] > (q_low - q_dif)) &
                   (data["Revenue"] < (q_hi + q_dif))]

s1 = pd.merge(data, df_filtered, how='inner')
```

Šalinamų elementų kiekis:

Employees: 36
Revenue: 0
Expenses: 0
Profit: 0
Growth: 0

Employees turėjo daugiausia atsiskyrėlių, tai gali indikuoti, kad per daug mažas koeficientas barjero. Taip pat kadangi nagrinėjamos Fortune 500 kompanijos viršutinė ribą gali būti padidinta, nes nagrinėjamos didžiausio įmonės. Todėl employees turėtų būti nagrinėjamas papildomai.

Duomenų normavimas

Min-Max Normavimas

Min-Max normavimas buvo atliktas naudojant sklearn.preprocessing

```
scaler = MinMaxScaler()
minmax_scaled = s1.copy()
minmax_scaled[["Revenue", "Expenses", "Profit", "Growth", "Employees"]] = scaler.fit_transform(
    s1[["Revenue", "Expenses", "Profit", "Growth", "Employees"]])
```

ir gauti rezultatai:

	Employees	Revenue	Expenses	Profit	Growth
count	459.000000	459.000000	459.000000	459.000000	459.000000
mean	0.199027	0.453909	0.439685	0.356515	0.531458
std	0.205348	0.157664	0.214205	0.209048	0.209382
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.062189	0.343678	0.283465	0.179506	0.333333
50%	0.121891	0.445237	0.440128	0.357079	0.575758
75%	0.253731	0.564660	0.596619	0.508046	0.696970
max	1.000000	1.000000	1.000000	1.000000	1.000000

Normavimas pagal vidurkį ir dispersiją

```
x = s1.loc[:, scl_names].values
x = StandardScaler().fit_transform(x)

act_data = s1[["Revenue", "Expenses", "Profit", "Growth", "Employees"]]
mapper = DataFrameMapper([(act_data.columns, StandardScaler())])

scaled_features = mapper.fit_transform(act_data.copy(), 4)
scaled_features_df = pd.DataFrame(
    scaled_features, index=act_data.index, columns=act_data.columns)
std_scaled = s1.copy()
std_scaled[["Revenue", "Expenses", "Profit", "Growth", "Employees"]] =
    scaled_features_df[["Revenue", "Expenses", "Profit", "Growth", "Employees"]]
```

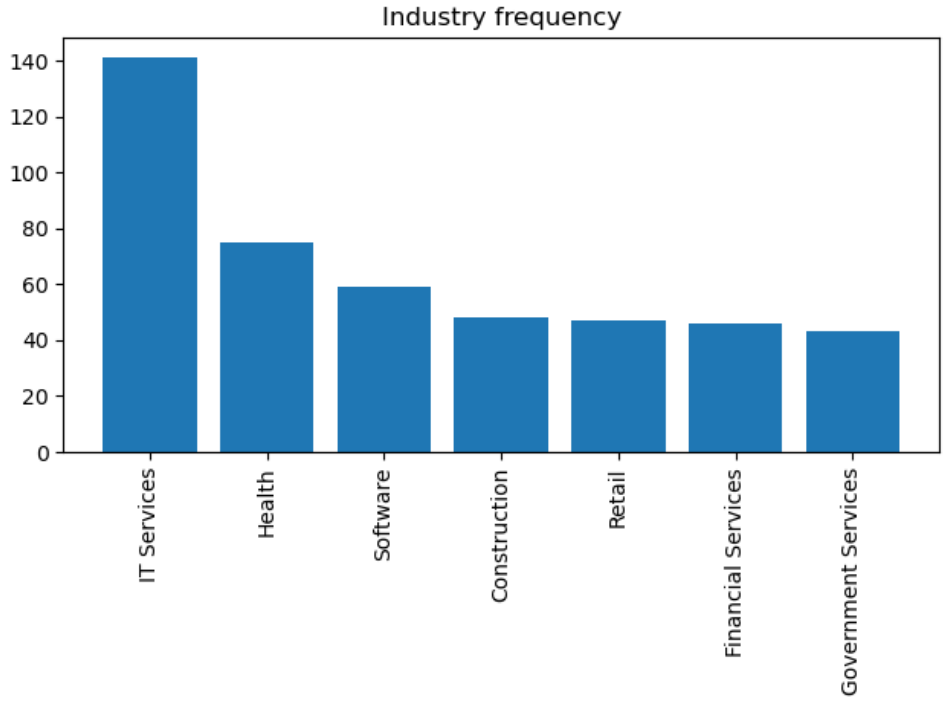
ir gauti rezultatai:

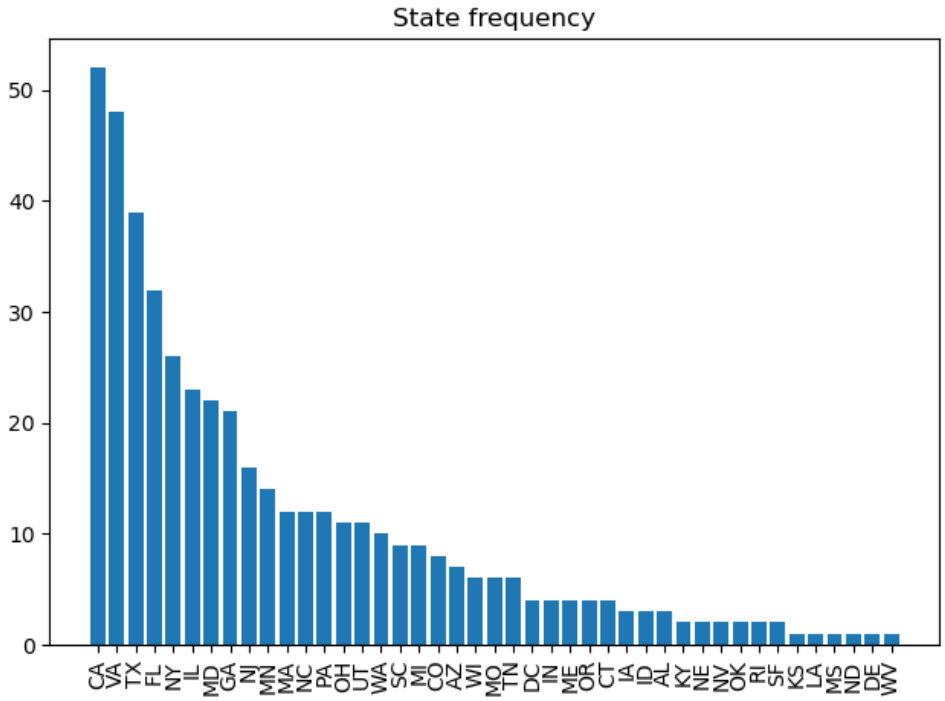
	Employees	Revenue	Expenses	Profit	Growth
count	4.590000e+02	4.590000e+02	4.590000e+02	4.590000e+02	4.590000e+02
mean	7.272027e-09	-4.155444e-09	-4.155444e-09	-1.662177e-08	6.233166e-09
std	1.001091e+00	1.001091e+00	1.001091e+00	1.001091e+00	1.001091e+00
min	-9.702740e-01	-2.882099e+00	-2.054878e+00	-1.707284e+00	-2.540996e+00
25%	-6.670964e-01	-6.999158e-01	-7.300987e-01	-8.476647e-01	-9.472709e-01
50%	-3.760459e-01	-5.506565e-02	2.072976e-03	2.703224e-03	2.118023e-01
75%	2.666906e-01	7.032116e-01	7.334369e-01	7.256549e-01	7.913389e-01
max	3.904822e+00	3.467409e+00	2.618649e+00	3.081531e+00	2.240180e+00

Vizualizacijos

Bendri grafikai

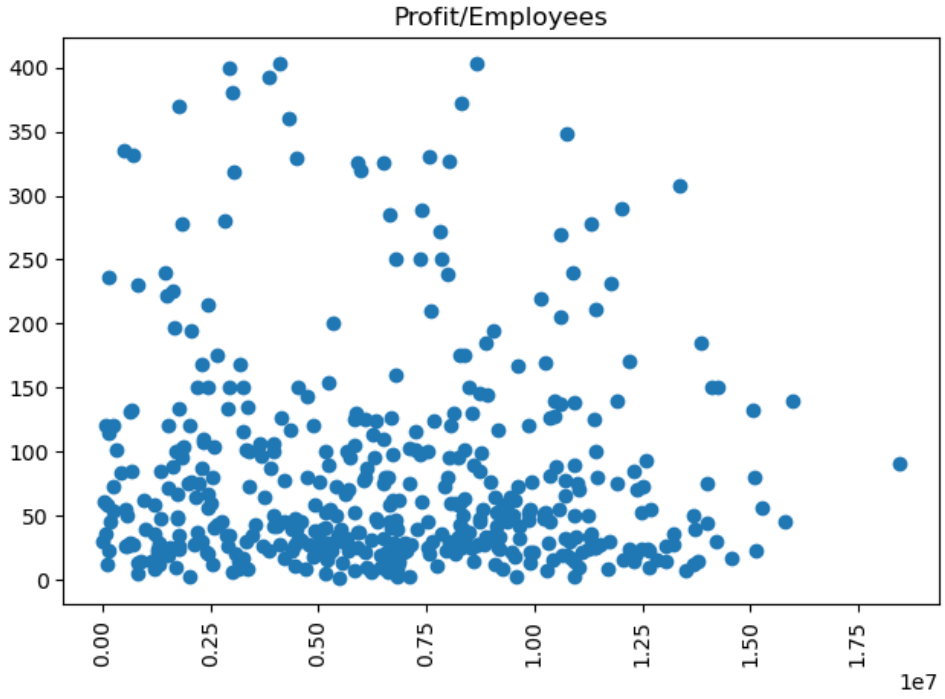
Darbuotojų kiekis pagal Industry ir State



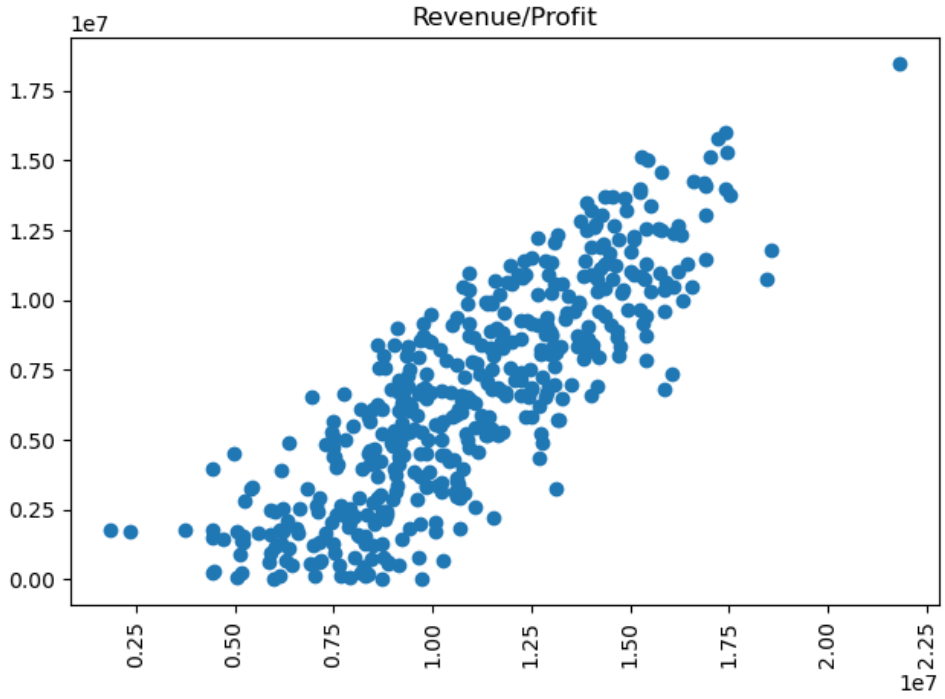


Šiuose grafikuose galime matyt, jog daugiausiai Employees yra IT Services Industry ir CA State, tai yra tikriausiai dėl Silicon valley.

Kiekybinių priklausomybė



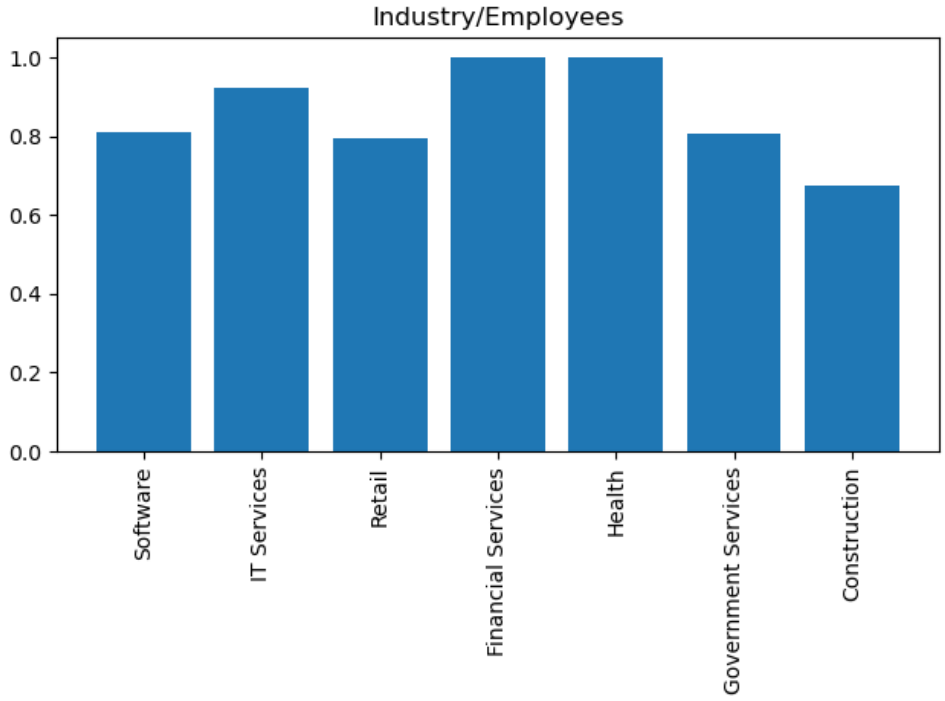
Iš šio grafiko galime matyti, jog Profit nepriklauso nuo Employees, nes net ir nedaug darbuotojų turinčios įmonės turi aukštą profit.



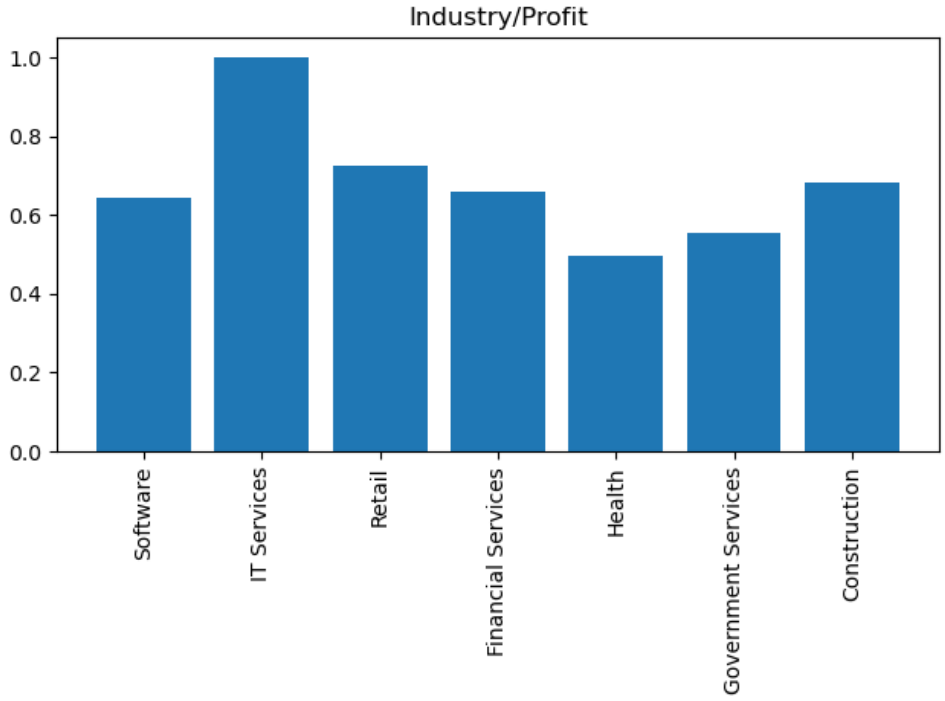
Iš šio grafiko galime matyti, jog Profit yra susiję su Revenue - kuo didesnė apyvartą tuo didesnis ir pelnas.

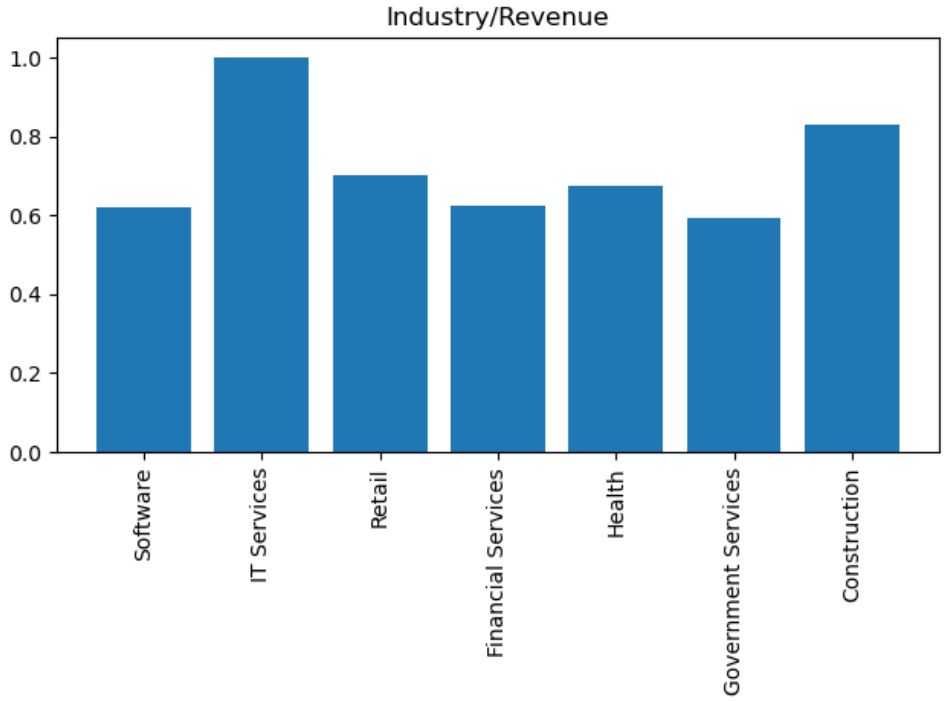
Min-Max normuoti grafikai

Pagal Industry



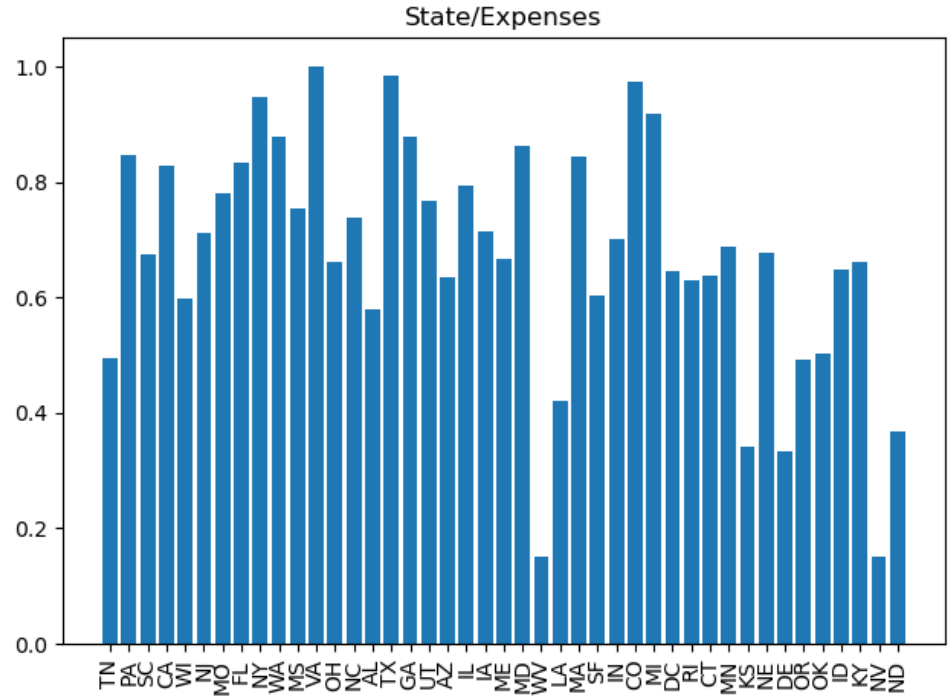
Matome kad industrijos Employees maksimumai yra pakankamai panašus.





Industry Profit ir Revenue maksimumus dominuoja IT Services. O blogiausiai pasirodo Health ir Government Services.

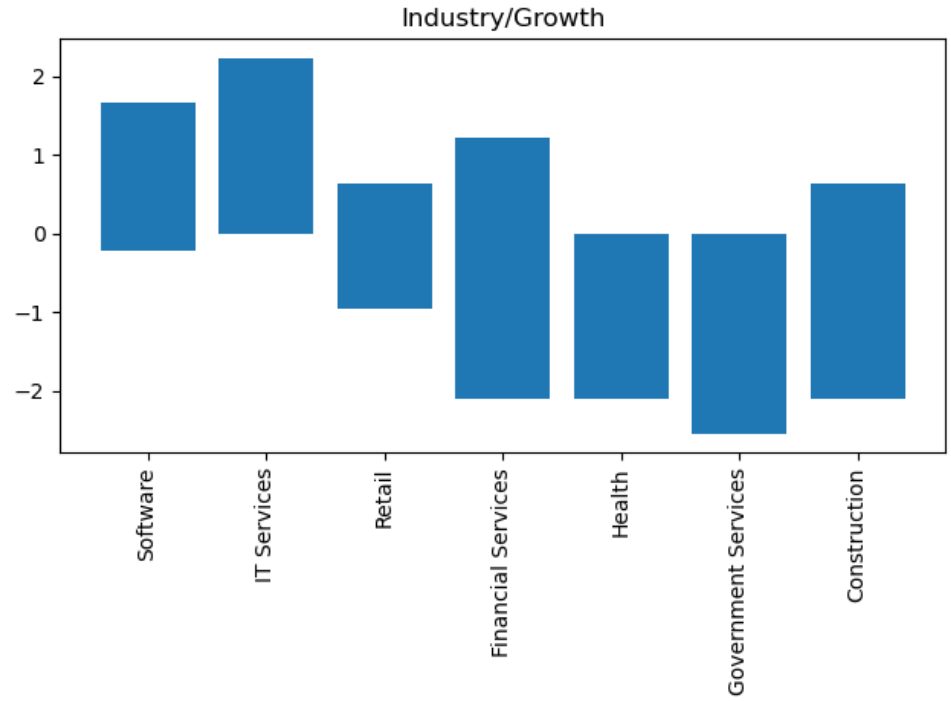
Pagal State



Matome kad State Expenses maksimumus yra stipriai mažesnis WV (~0.2) ir NV (~0.2) State, kas reiškia, kad ten pigiausia kurti įmonę.

Vidurkiu ir dispersija normuoti grafikai

Pagal Industry



Growth yra didžiausias IT services (>2), o mažiausias Government services (<-2).

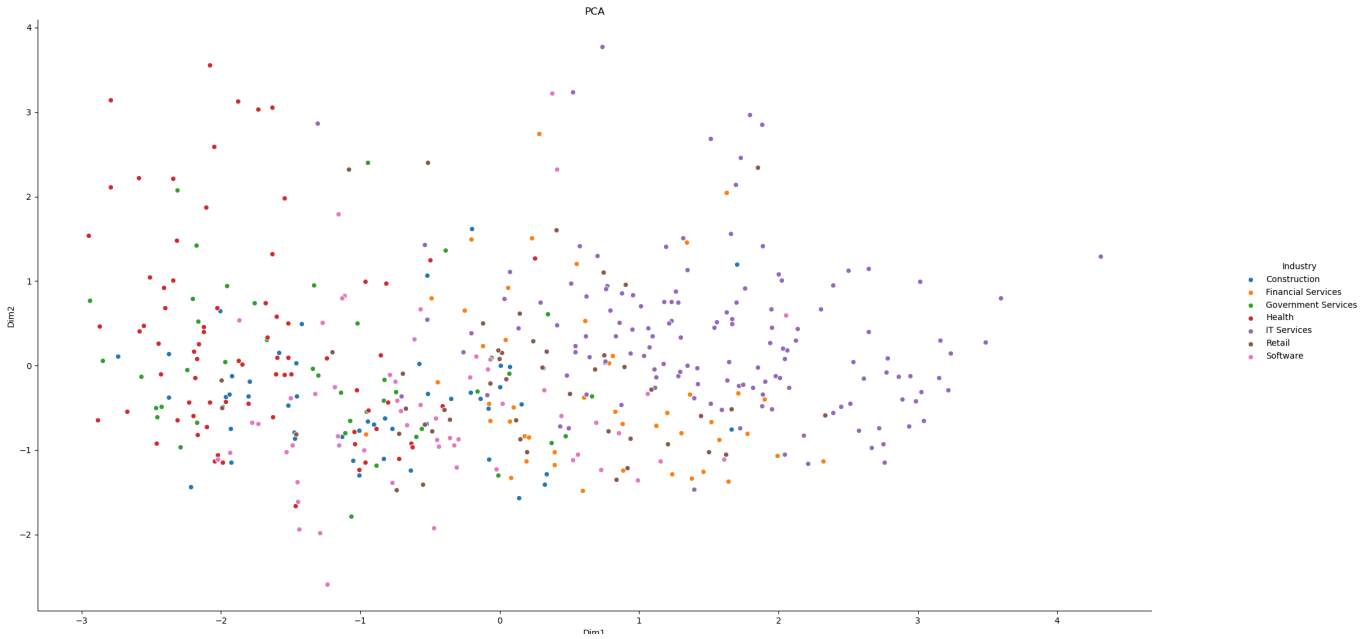
Pagal State

Nenagrinėjama, nes per mažai duomenų, kad vidurkis būtų tinkamas indikatorius.

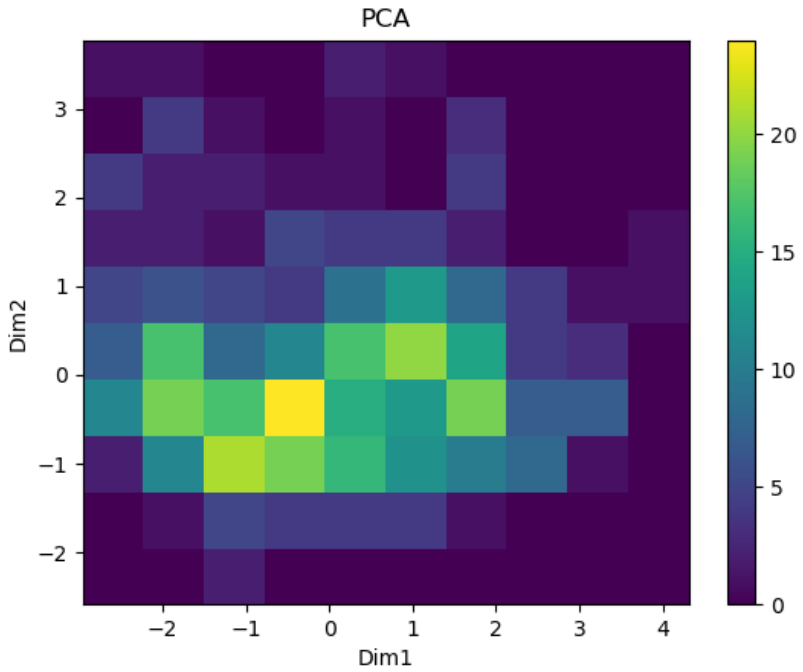
PCA ir MDS algoritmas

Daugiamačiam nagrinėjimui buvo pasirinktas PCA algoritmas naudojant sklearn.decomposition ir seaborn vizualizacijai:

```
#x - Normuoti duomenys pagal vidurkį ir dispersiją kiekybiniai
#std_scaled - normuoti visi duomenys
pca = PCA(n_components=2)
principalComponents = pca.fit_transform(x)
principal_Df = pd.DataFrame(data=principalComponents, columns=['Dim1', 'Dim2'])
principal_Df['Industry'] = std_scaled['Industry']
sns.pairplot(x_vars=["Dim1"], y_vars=["Dim2"], data=principal_Df, hue="Industry")
```



Pagal taškine diagramą galime matyti, jog vienoje plokštumos pusėje yra IT Services ir Financial Sector, o kitoje Government Services ir Health. Pagal tai galime teigti, kad šios dvi grupės skiriasi pagal nagrinėjamus rodiklius.



Pagal šią diagramą galime matyti kad turime du klasterius, tačiau jie nėra toli vienas nuo kito, pagal ką galime teigti, jog vidutiniškai rodikliai yra panašus (nėra kategoriškai besiskiriantys)

Koreliacija

Koreliacija gaunama naudojant Pandas:

```
data.corr(method='pearson')
```

	Employees	Revenue	Expenses	Profit	Growth
Employees	1.000000	-0.022411	0.063000	-0.052620	-0.083790
Revenue	-0.022411	1.000000	-0.033675	0.835522	0.448901
Expenses	0.063000	-0.033675	1.000000	-0.576776	-0.250309
Profit	-0.052620	0.835522	-0.576776	1.000000	0.502986
Growth	-0.083790	0.448901	-0.250309	0.502986	1.000000

Iš lentelės matome, kad Revenue ir Profit tikrai stipriai susiję teigiamai (0.835522). Taip pat matome, kad Profit ir Expenses susiję neigiamai (-0.576776). Iš ko galime spręsti, jog norint pasiekti didžiausia Profit turime turėti kuo didesne Revenue ir kuo mažesnius Expenses.

Išvados

Darbo eigoje gaunant tarpinius rezultatus paaiškėjo tam tikri dalykai apie nagrinėjamą duomenų aibę:

1. Reikia papildomos analizės dėl darbuotojų kiekio išorinio barjero, kadangi nagrinėjamos didžiausios įmonės.
2. Pagal Dažnio diagramas matome, jog daugiausiai Employees yra IT Services Industry ir CA State.
3. Profit nepriklauso nuo Employees, nes net ir nedaug darbuotojų turinčios įmonės turi aukštą pelną.
4. Profit yra stipriai susijęs su Revenue - kuo didesnė apyvartą tuo didesnis ir pelnas. Šiuos duomenys galime matyti pagal taškinės diagramos tendencijas ir pagal koreliacijos lentelę.
5. Pagal Min-Max normuotų duomenų stulpelinę diagramą matome, jog Profit ir Revenue maksimumus dominuoja IT Services Industry.
6. Pagal normuotų duomenų stulpelinę diagramą matome, kad Industry Employees kiekių viršūnės yra panašios - įmonės vidutiniškai turi vienodus kiekius darbuotojų.
7. Pagal stulpelinę diagramą matome WV ir NV State įsteigtos įmonės turi mažiausius Expenses.
8. Norint turėti aukšta pelną reikia didelės apyvartos ir mažų išlaidų ir tam geriausiai tinka IT service industrija didžiausiam Revenue ir WV arba NV State mažiausiems kaštas.
9. Iš PCA diagramos matome, jog IT Services ir Financial Financial Sector sudaro grupę ir Government Services ir Health sudaro kitą grupę. Kadangi žinome, jog didžiausia pelną turi IT Services, o mažiausią augimą Government Services, galime daryti išvadas, jog Financial Sector taip pat yra daugiau pelningas, o Health mažiau pelningas.