Analiza modela

In [ ]:
```
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
```

Prvo čitamo tablicu s podacima o rezultatima naših modelima.

In [ ]:
```
df = pd.read_csv('scores.csv')
df.head()
```

Out[ ]:

	train_sample_size	N	n_components	rho	linreg	pcr	pls
0	5000.0	500.0	5.0	0.01	0.999997	0.017560	0.999945
1	5000.0	500.0	50.0	0.01	0.999997	0.103794	0.999997
2	5000.0	500.0	125.0	0.01	0.999997	0.264234	0.999997
3	5000.0	500.0	250.0	0.01	0.999997	0.544558	0.999997
4	2500.0	500.0	5.0	0.01	0.999996	0.019106	0.999126

Za potrebe crtanja grafova pivotirat ćemo tablicu tako da su imena modela vrijednosti stupca *model*, a njihove R2 metrike sve u istom stupcu.

In [ ]:
```
df = pd.read_csv('scores.csv')
df = df.melt(['train_sample_size', 'N', 'n_components', 'rho'], var_name='mo
df.head()
```

Out[ ]:

	train_sample_size	N	n_components	rho	model	R2
0	5000.0	500.0	5.0	0.01	linreg	0.999997
1	5000.0	500.0	50.0	0.01	linreg	0.999997
2	5000.0	500.0	125.0	0.01	linreg	0.999997
3	5000.0	500.0	250.0	0.01	linreg	0.999997
4	2500.0	500.0	5.0	0.01	linreg	0.999996

Potom dodajemo informacije o omjeru dimenzije N naprema veličini training set-a i broju komponenti koje su korištenje u PCR-u i PLS-u u stupce "train_sample_size_N_ratio" i "component_N_ratio" redom.

In [ ]:
```
df['train_sample_size_N_ratio'] = df.apply(lambda x: x.train_sample_size / x
df['component_N_ratio'] = df.apply(lambda x: x.n_components / x.N, axis=1)
df.head()
```

Out[ ]:

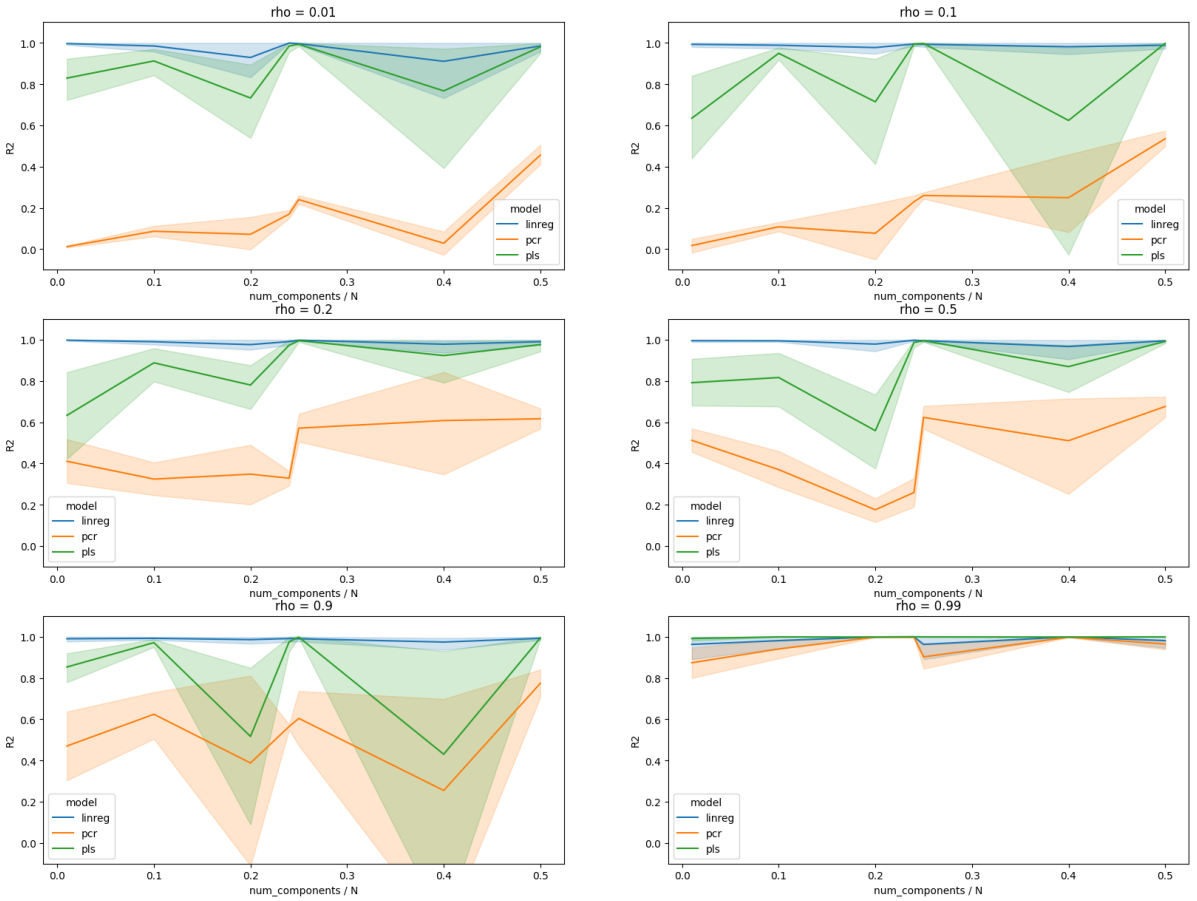
	train_sample_size	N	n_components	rho	model	R2	train_sample_size_N_rat
0	5000.0	500.0	5.0	0.01	linreg	0.999997	10
1	5000.0	500.0	50.0	0.01	linreg	0.999997	10
2	5000.0	500.0	125.0	0.01	linreg	0.999997	10
3	5000.0	500.0	250.0	0.01	linreg	0.999997	10
4	2500.0	500.0	5.0	0.01	linreg	0.999996	5

Sad cemo grafički prikazati kako R2 ovisi o gore uvedenim omjerima. Nacrtat cemo po jedan graf za svaku vrijednost kovarijance ρ među kovarijatama.

In [ ]:
```
plt.figure(figsize=(20, 20))

plt_y = 2
plt_x = df.rho.nunique() // plt_y + 1

for idx, rho in enumerate(df.rho.unique()):
    df_rho = df[df.rho == rho]
    plt.subplot(plt_x, plt_y, idx+1)
    sns.lineplot(x='component_N_ratio', y='R2', hue='model', data=df_rho)
    plt.title(f"rho = {rho}")
    plt.ylim(-0.1, 1.1)
    plt.xlabel('num_components / N')
```

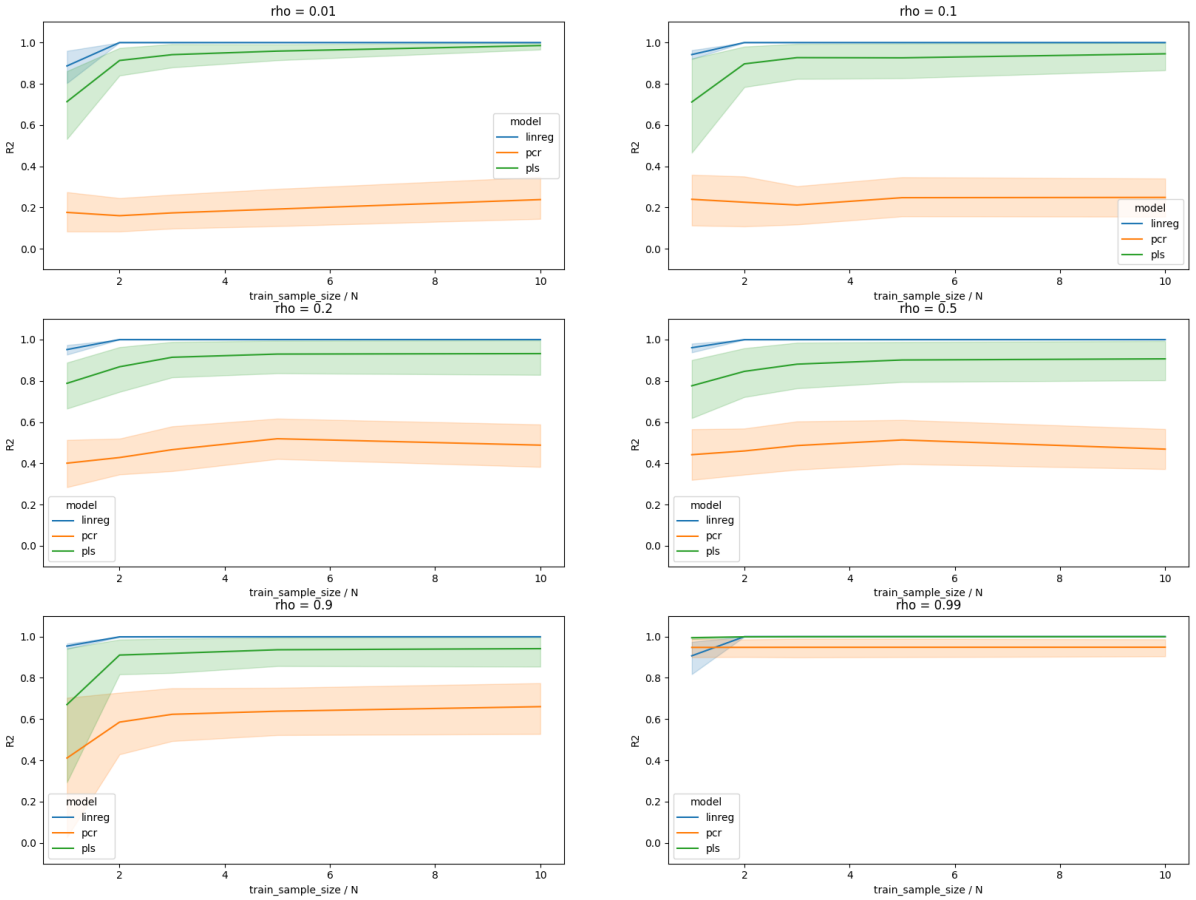


In [ ]:
```
plt.figure(figsize=(20, 20))

plt_y = 2
plt_x = df.rho.nunique() // plt_y + 1

for idx, rho in enumerate(df.rho.unique()):
    df_rho = df[df.rho == rho]
```

```
    plt.subplot(plt_x, plt_y, idx+1)
    sns.lineplot(x='train_sample_size_N_ratio', y='R2', hue='model', data=df
    plt.title(f"rho = {rho}")
    plt.ylim(-0.1, 1.1)
    plt.xlabel('train_sample_size / N')
```

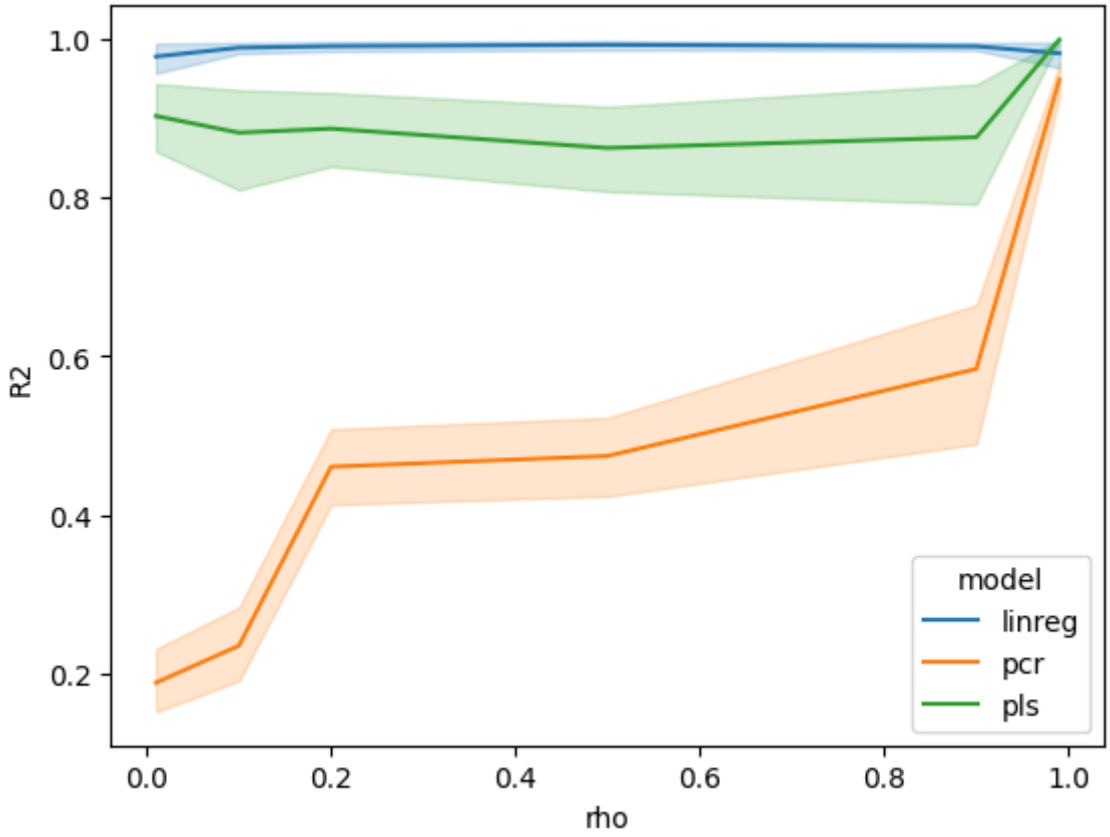


Među gornjim grafovima je zanimljivo uočiti kako je u slučaju najviše korelacije $\rho = 0.99$ linearna regresija najlošija među metodama ukoliko imamo malen uzorak.

Sljedeći graf prikazuje koliko metode u prosjeku ovise o koeficijentu korelacije ρ. Ovdje se jasno vidi da PCR više ovisi o veličini korelacije nego PLS. Također je zanimljivo da kod najveće razine korelacije $\rho = 0.99$ PLS u prosjeku čak mrvu bolje predviđa od linearne regresije.

In [ ]:
```
sns.lineplot(x='rho', y='R2', data = df, hue='model')
```

Out[ ]:
```
<AxesSubplot: xlabel='rho', ylabel='R2'>
```


1.0
0.8
0.6
0.4
0.2
R2
0.0
0.2
0.4
0.6
0.8
1.0
rho
model
linreg
pcr
pls