

```
In [4]: import pandas as pd
import numpy as np

raw = pd.read_csv("boston.txt", sep=r"\s+", skiprows=22, header=None)
X = np.hstack([raw.values[:, 2:], raw.values[1:, 2:]])
y = raw.values[1:, 2]
columns = ["crim", "zn", "indus", "chas", "nox", "rm", "age",
           "dis", "rad", "tax", "ptratio", "b", "lstat"]
df = pd.DataFrame(X, columns=columns)
df["medv"] = y
```

a

```
In [5]: import matplotlib.pyplot as plt
from seaborn import pairplot
import statsmodels.api as sm

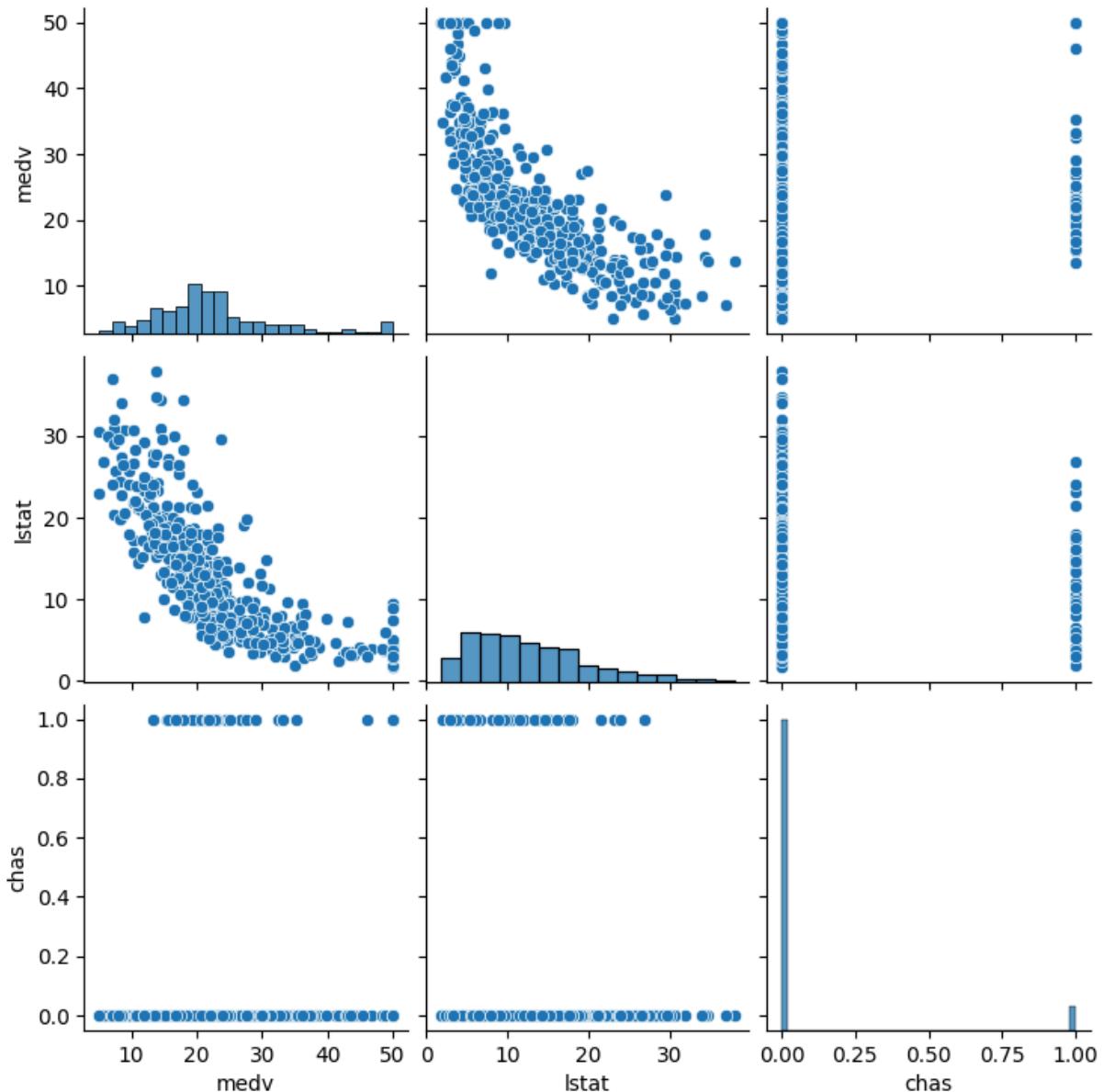
pairplot(df[["medv", "lstat", "chas"]])
X = sm.add_constant(df[["lstat", "chas"]])
model = sm.OLS(df["medv"], X).fit()
print(model.summary())
```

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.563
Model:	OLS	Adj. R-squared:	0.561
Method:	Least Squares	F-statistic:	323.4
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	4.93e-91
Time:	12:22:43	Log-Likelihood:	-1631.1
No. Observations:	506	AIC:	3268.
Df Residuals:	503	BIC:	3281.
Df Model:	2		
Covariance Type:	nonrobust		
<hr/>			
	coef	std err	t
const	34.0941	0.561	60.809
lstat	-0.9406	0.038	-24.729
chas	4.9200	1.069	4.601
<hr/>			
Omnibus:	131.896	Durbin-Watson:	0.975
Prob(Omnibus):	0.000	Jarque-Bera (JB):	275.510
Skew:	1.406	Prob(JB):	1.49e-60
Kurtosis:	5.272	Cond. No.	57.8
<hr/>			

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.



The pair plots indicate a non linear relationship between `lstat` and `medv`, `medv` drops quickly for low-mid `lstat`, and then flattens out. Usually taking the log, square root, or inverse of `lstat` can help fix this. For this problem, we will use a log transformation.

Intercept: 34.0941, SE 0.561, p < 0.0001

This is the predicted `medv` for an `lstat` and `chas` value of 0. This isn't meaningful realistically as `lstat` 0 isn't possible, but this anchors the regression line. The value is statistically significant and well estimated from the low SE and p value.

`lstat` coefficient: -0.9406, SE 0.038, p < 0.0001

Holding `chas` constant, a 1 percentage point increase in `lstat` is attributed to about a \$940.6 decrease in `medv`. The value is statistically significant and well estimated from the low SE and p value.

`chas` coefficient: 4.92, SE 1.069, p < 0.0001 Holding `lstat` constant, properties that border the Charles River are attributed to a \$4,920 increase in `medv`. The value is

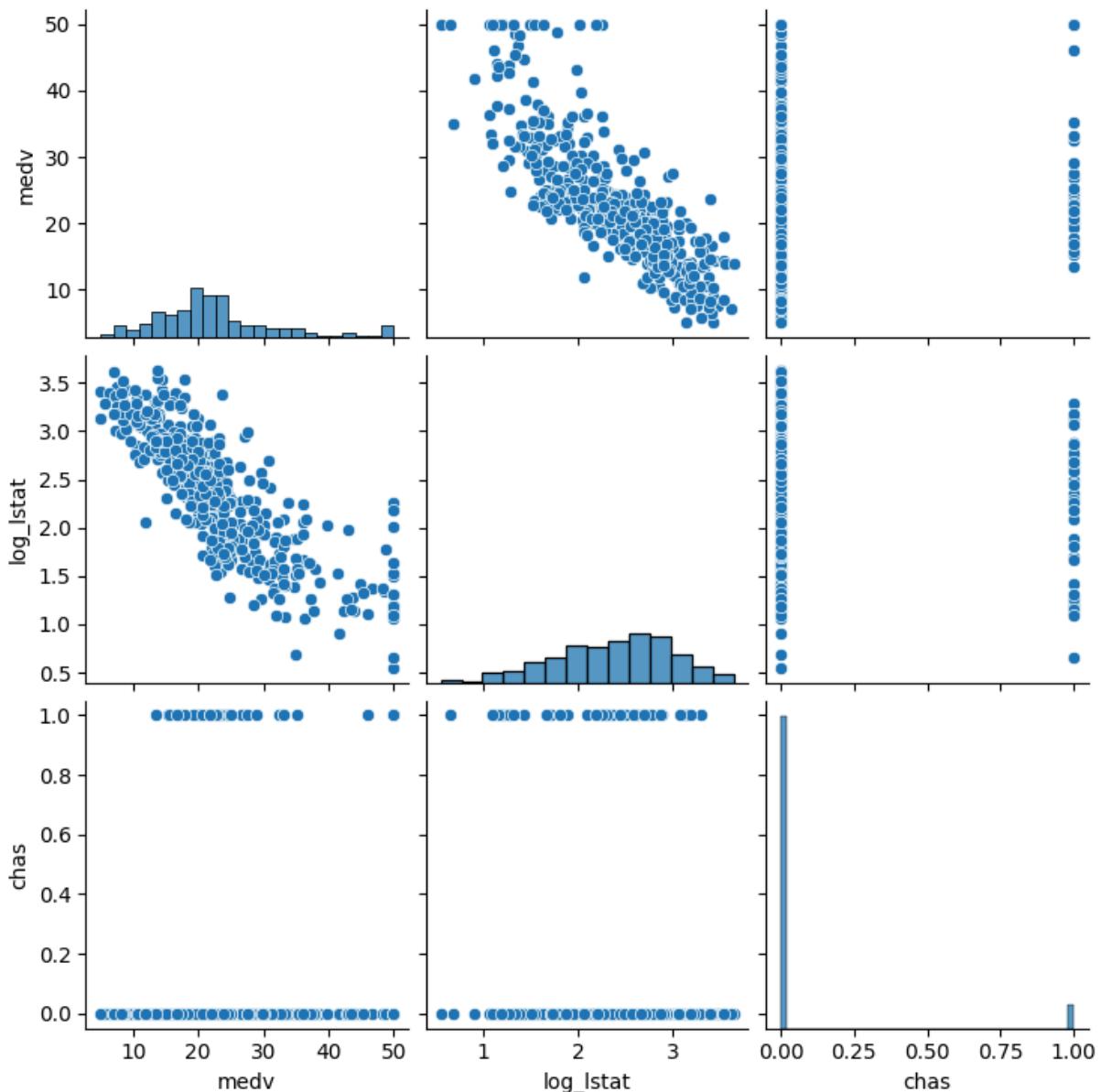
statistically significant and well estimated from the low SE and p value.

```
In [6]: df["log_lstat"] = np.log(df["lstat"])
pairplot(df[["medv", "log_lstat", "chas"]])
X = sm.add_constant(df[["log_lstat", "chas"]])
model = sm.OLS(df["medv"], X).fit()
print(model.summary())
```

OLS Regression Results						
		medv	R-squared:	0.678		
Dep. Variable:		OLS	Adj. R-squared:	0.677		
Model:		Least Squares	F-statistic:	530.1		
Date:	Fri, 13 Feb 2026		Prob (F-statistic):	1.43e-124		
Time:	12:22:54		Log-Likelihood:	-1553.4		
No. Observations:	506		AIC:	3113.		
Df Residuals:	503		BIC:	3125.		
Df Model:	2					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
const	51.5250	0.956	53.899	0.000	49.647	53.403
log_lstat	-12.3500	0.388	-31.814	0.000	-13.113	-11.587
chas	4.1819	0.918	4.554	0.000	2.378	5.986
Omnibus:	126.805		Durbin-Watson:	0.990		
Prob(Omnibus):	0.000		Jarque-Bera (JB):	335.582		
Skew:	1.230		Prob(JB):	1.35e-73		
Kurtosis:	6.140		Cond. No.	11.9		

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.



Intercept: 51.5250, SE 0.956, p < 0.0001

This is the predicted `medv` for an `lstat` and `chas` value of 0. This isn't meaningful realistically as `lstat` 0 isn't possible, but this anchors the regression line. The value is statistically significant and well estimated from the low SE and p value.

log_lstat coefficient: -12.3500, SE 0.388, p < 0.0001

Holding `chas` constant, a 1 percentage point increase in `lstat` is attributed to about a \$123.50 decrease in `medv`. The value is statistically significant and well estimated from the low SE and p value.

chas coefficient: 4.1819, SE .0918, p < 0.0001 Holding `lstat` constant, properties that border the Charles River are attributed to a \$4,181.90 increase in `medv`. The value is statistically significant and well estimated from the low SE and p value.

The adjusted R² of the model is much higher than the original model, already showing an improvement. From the pairplot, we can see the relationship between `log_lstat` and

`medv` is linear now, showing the log transformation helped.

b

```
In [7]: df["loglstat_chas"] = df["log_lstat"] * df["chas"]
X = sm.add_constant(df[["log_lstat", "chas", "loglstat_chas"]])
model_interaction = sm.OLS(df["medv"], X).fit()
print(model_interaction.summary())
```

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.680			
Model:	OLS	Adj. R-squared:	0.678			
Method:	Least Squares	F-statistic:	354.8			
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	1.35e-123			
Time:	12:22:57	Log-Likelihood:	-1552.4			
No. Observations:	506	AIC:	3113.			
Df Residuals:	502	BIC:	3130.			
Df Model:	3					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
const	51.0995	1.000	51.075	0.000	49.134	53.065
log_lstat	-12.1715	0.408	-29.868	0.000	-12.972	-11.371
chas	8.3909	3.090	2.715	0.007	2.320	14.462
loglstat_chas	-1.8922	1.327	-1.426	0.154	-4.498	0.714
Omnibus:	128.919	Durbin-Watson:	1.002			
Prob(Omnibus):	0.000	Jarque-Bera (JB):	351.383			
Skew:	1.239	Prob(JB):	4.99e-77			
Kurtosis:	6.245	Cond. No.	38.2			

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Intercept: 51.0995, SE 1, p < 0.0001

This is the predicted `medv` for an `lstat` and `chas` value of 0. This isn't meaningful realistically as `lstat` 0 isn't possible, but this anchors the regression line. The value is statistically significant and well estimated from the low SE and p value.

log_lstat coefficient: -12.1715, SE 0.408, p < 0.0001

Holding `chas` constant, a 1 percentage point increase in `lstat` is attributed to about a \$121.72 decrease in `medv`. The value is statistically significant and well estimated from the low SE and p value. The coefficient is slightly smaller than the model without the interaction term.

chas coefficient: 8.3909, SE 3.090, p 0.007 Holding `lstat` constant, properties that border the Charles River are attributed to a \$9,825.10 increase in `medv`. The value is

statistically significant and well estimated from the low SE and p value. The coefficient is significantly larger than the model without the interaction term.

lstat:chas coefficient: -1.8922, SE 1.327, p 0.154

The negative effect of `lstat` is stronger for properties bordering the Charles River. The value is not statistically significant nor well estimated from the high SE and p value. Specifically, for properties that border the Charles River, a 1 percentage point increase in `lstat` decreased `medv` by \$14.06.

Given that the new R squared value of this model 0.680 is barely an improvement from the previous model of 0.678, and the interaction term is not statistically significant, interactions do not improve the model.

C

```
In [8]: df0 = df[df["chas"] == 0]
df1 = df[df["chas"] == 1]

X0 = sm.add_constant(df0["log_lstat"])
model0 = sm.OLS(df0["medv"], X0).fit()
print(model0.summary())

X1 = sm.add_constant(df1["log_lstat"])
model1 = sm.OLS(df1["medv"], X1).fit()
print(model1.summary())
```

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.664
Model:	OLS	Adj. R-squared:	0.663
Method:	Least Squares	F-statistic:	925.6
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	4.68e-113
Time:	12:23:00	Log-Likelihood:	-1437.2
No. Observations:	471	AIC:	2878.
Df Residuals:	469	BIC:	2887.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	51.0995	0.982	52.025	0.000	49.169	53.030
log_lstat	-12.1715	0.400	-30.424	0.000	-12.958	-11.385

Omnibus:	127.277	Durbin-Watson:	0.996
Prob(Omnibus):	0.000	Jarque-Bera (JB):	347.256
Skew:	1.306	Prob(JB):	3.93e-76
Kurtosis:	6.297	Cond. No.	11.8

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.713
Model:	OLS	Adj. R-squared:	0.704
Method:	Least Squares	F-statistic:	81.96
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	1.84e-10
Time:	12:23:00	Log-Likelihood:	-113.75
No. Observations:	35	AIC:	231.5
Df Residuals:	33	BIC:	234.6
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	59.4904	3.598	16.536	0.000	52.171	66.810
log_lstat	-14.0637	1.553	-9.053	0.000	-17.224	-10.903

Omnibus:	9.133	Durbin-Watson:	1.242
Prob(Omnibus):	0.010	Jarque-Bera (JB):	10.478
Skew:	0.686	Prob(JB):	0.00531
Kurtosis:	5.303	Cond. No.	8.99

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Model	n	R^2	Adj R^2	AIC	BIC
Interaction	506	0.680	0.678	3113	3130

Model	n	R^2	Adj R^2	AIC	BIC
chas = 0	471	0.664	0.664	2878	2887
chas = 1	35	0.713	0.704	231.5	234.6

Note that the slopes for each `chas` specific model are the same as the indicated slopes in the interaction model when accounting for the interaction term. Additionally, the `chas = 1` model only has 35 samples. Thus, we prefer the original log `lstat` model due to it using all 506 observations, and more importantly, because the interaction term is not statistically significant, using a common slope reduces variance, increases efficiency (in future predictions), simplifies interpretation, and avoids overfitting the smaller group (`chas = 1`). We can see this already occurring as the R^2 for the `chas = 1` model is higher.

d

```
In [9]: df_restrict = df[df["rad"] <= 8]

X_num = sm.add_constant(df_restrict[["log_lstat", "rad"]].astype(float))
model_num = sm.OLS(df_restrict["medv"].astype(float), X_num).fit()
print(model_num.summary())

rad_dummies = pd.get_dummies(df_restrict["rad"], prefix="rad", drop_first=True)
X_cat = sm.add_constant(pd.concat([df_restrict["log_lstat"].astype(float), rad_dumm
model_cat = sm.OLS(df_restrict["medv"].astype(float), X_cat).fit()
print(model_cat.summary())
```

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.606
Model:	OLS	Adj. R-squared:	0.603
Method:	Least Squares	F-statistic:	284.8
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	1.12e-75
Time:	12:23:08	Log-Likelihood:	-1153.1
No. Observations:	374	AIC:	2312.
Df Residuals:	371	BIC:	2324.
Df Model:	2		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	48.4863	1.337	36.263	0.000	45.857	51.115
log_lstat	-11.6251	0.489	-23.757	0.000	-12.587	-10.663
rad	0.4140	0.168	2.462	0.014	0.083	0.745

Omnibus:	71.571	Durbin-Watson:	0.945
Prob(Omnibus):	0.000	Jarque-Bera (JB):	125.233
Skew:	1.088	Prob(JB):	6.40e-28
Kurtosis:	4.816	Cond. No.	26.9

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

OLS Regression Results

Dep. Variable:	medv	R-squared:	0.634
Model:	OLS	Adj. R-squared:	0.626
Method:	Least Squares	F-statistic:	78.89
Date:	Fri, 13 Feb 2026	Prob (F-statistic):	7.15e-75
Time:	12:23:08	Log-Likelihood:	-1139.4
No. Observations:	374	AIC:	2297.
Df Residuals:	365	BIC:	2332.
Df Model:	8		
Covariance Type:	nonrobust		

	coef	std err	t	P> t	[0.025	0.975]
const	46.2099	1.501	30.788	0.000	43.258	49.161
log_lstat	-11.2540	0.495	-22.715	0.000	-12.228	-10.280
rad_2.0	4.4640	1.563	2.857	0.005	1.391	7.537
rad_3.0	4.7632	1.425	3.344	0.001	1.962	7.565
rad_4.0	1.6991	1.270	1.338	0.182	-0.797	4.196
rad_5.0	4.3335	1.255	3.452	0.001	1.865	6.802
rad_6.0	2.4319	1.554	1.565	0.118	-0.624	5.488
rad_7.0	3.1987	1.700	1.881	0.061	-0.145	6.542
rad_8.0	5.9636	1.560	3.822	0.000	2.895	9.032

Omnibus:	77.028	Durbin-Watson:	1.016
Prob(Omnibus):	0.000	Jarque-Bera (JB):	143.714
Skew:	1.131	Prob(JB):	6.21e-32
Kurtosis:	5.027	Cond. No.	31.9

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Model	n	R^2	Adj R^2	AIC	BIC
Numerical rad	374	0.606	0.603	2312	2324
Categorical rad	374	0.634	0.626	2297	2332

Using `rad` as categorical significantly improves R^2 by 0.23. The AIC strongly favors the categorical specification, as the delta of 15 shows that allowing non-linear, level-specific effects of `rad` improves the fit. BIC slightly favors numerical specification however, as `rad` is heavily penalized with extra parameters as a categorical variable. The numerical model indicates that the `rad` coefficient is statistically significant. This does make sense, however it also implies that the movement from different `rads` has the same effect, which isn't necessarily realistic or true. The categorical model on the other hand, shows that different `rads` have varying statistical significance levels. This matches how `rad` is constructed realistically as it is an index, not a continuous economic quantity. Thus overall, we prefer the categorical specification due to higher adjusted R^2 and better representation of the predictor variables.

e

```
In [10]: b3 = model_cat.params["rad_3.0"]
b4 = model_cat.params["rad_4.0"]
diff = b3 - b4
print(f"Estimated difference in expected value of medv between houses w/ rad = 3 and rad = 4: {diff}")

cov = model_cat.cov_params()
var_diff = (
    cov.loc["rad_3.0", "rad_3.0"]
    + cov.loc["rad_4.0", "rad_4.0"]
    - 2 * cov.loc["rad_3.0", "rad_4.0"]
)
se_diff = np.sqrt(var_diff)
print(f"Estimated standard error of the difference: {se_diff}")
```

Estimated difference in expected value of medv between houses w/ rad = 3 and rad = 4: 3.0640954476181186

Estimated standard error of the difference: 0.9816951849191523