统计学习 Homework 1

12112627 李乐平

Questions chosen:
1. 2. 3. 6

1. We first compute the posterior distribution: i.e. $Pr(\vec{\beta}|\vec{y}) \propto Pr(\vec{\beta})Pr(\vec{y}|\vec{\beta})$.

$$\propto e^{-\frac{\|\vec{y}-X\vec{\beta}\|^2}{2\sigma^2}-\frac{\|\vec{\beta}\|^2}{2\tau}}$$

$$\therefore -\ln Pr(\vec{\beta}|\vec{y}) = \frac{\|\vec{y}-X\vec{\beta}\|^2}{2\sigma^2}+\frac{\|\vec{\beta}\|^2}{2\tau}+C$$

where C is a constant irrelevant with $\vec{\beta}$.

and this expression is the same with the target function in ridge regression if ignoring the constant term, ~~showing~~

By letting the first order derivative equal to 0, we can easily get when $\hat{\beta}$ equals the mode. i.e.

$\hat{\beta}=(\frac{\sigma^2}{\tau}I+X^TX)^{-1}X^Ty$, the target function is optimized. this result is equivalent to the ridge regression. if $\lambda=\frac{\sigma^2}{\tau}$, which is the relationship we want to describe.

Yet we can clearly see the kernel of $Pr(\vec{\beta}|\vec{y})$ is consistent with Gaussian distribution:

$$Pr(\vec{\beta}|\vec{y}) \propto e^{(\vec{\beta}-(X^TX+\frac{\sigma^2}{\tau}I)^{-1}X^T\vec{y})^T(X^TX+\frac{\sigma^2}{\tau}I)(\vec{\beta}-(X^TX+\frac{\sigma^2}{\tau}I)^{-1}X^T\vec{y})+C}.$$

Showing the mean is also $\vec{\mu}=(X^TX+\frac{\sigma^2}{\tau}I)^{-1}X^T\vec{y}=\hat{\beta}$.

Q.E.D.

2. When $\vec{\beta}$ is given, we know that

$$f(y_i)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(y_i-(\beta_0+\vec{x}_i^T\vec{\beta}))^2}{2\sigma^2}}$$

$$\therefore f(\vec{y}|\vec{\beta})=\frac{1}{(\sqrt{2\pi}\sigma)^N}e^{-\frac{\sum_{i=1}^N(y_i-(\beta_0+\vec{x}_i^T\vec{\beta}))^2}{2\sigma^2}}$$

Yet we also know

$$\pi(\vec{\beta})=\frac{1}{(\sqrt{2\pi}\tau)^p}e^{-\frac{\sum_{j=1}^p b_j^2}{2\tau^2}}$$

By Bayesian formula.

$$f(\vec{\beta}|\vec{y}) \propto \pi(\vec{\beta})\cdot f(\vec{y}|\vec{\beta}) = e^{-\frac{\sum_{i=1}^N(y_i-(\beta_0+\vec{x}_i^T\vec{\beta}))^2}{2\sigma^2}-\frac{\sum_{j=1}^p b_j^2}{2\tau^2}}.$$

(2. cont.)

$$\therefore -\ln f(\vec{\beta}|\vec{y}) = \frac{\sum_{i=1}^N(y_i-(\beta_0+\vec{x}_{ij}^T\vec{\beta}))^2}{2\sigma^2}+\frac{\sum_{j=1}^p\beta_j^2}{2\tau^2}+C.$$

$$\propto \sum_{i=1}^N(y_i-\beta_0-\sum_j x_{ij}\beta_j)^2+\lambda\sum_{j=1}^p\beta_j^2 \quad (\lambda=\frac{\sigma^2}{\tau^2})$$

Q.E.D.

3. Notice that the problem

$$\min_{\beta}\|\vec{y}-X\vec{\beta}\|^2+\lambda[\alpha\|\vec{\beta}\|_2^2+(1-\alpha)\|\vec{\beta}\|_1]$$

is equivalent to

$$\cancel{\min_{\beta}\left\|\begin{bmatrix}\vec{y}\\ \vec{0}_p\end{bmatrix}-\begin{bmatrix}X\\ \sqrt{\lambda\alpha}I_{p\times p}\end{bmatrix}\vec{\beta}\right\|^2}$$

$$\min_{\beta}\left\|\begin{bmatrix}\vec{y}\\ \vec{0}_p\end{bmatrix}-\begin{bmatrix}X\\ \sqrt{\lambda\alpha}I_{p\times p}\end{bmatrix}\vec{\beta}\right\|^2+\lambda(1-\alpha)\|\vec{\beta}\|_1,$$

which is a LASSO problem. Q.E.D.

6. The solution to the question 6 is performed as python script.

Statistical Learning Homework 1

12112627 李乐平

Question 6

Reproduce prostate cancer example, using methods including LSE, LASSO, Ridge Regression and Elastic Net.

Solution: In this section, I reproduced the result of Table 3-1, Table 3-2 and Table 3-3 in ESL. The script is as follows.

In [1]:
```
# !python -m pip install --user --upgrade seaborn
# !python -m pip install --user --upgrade statsmodels
```

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.api as sm

from sklearn.linear_model import *
from sklearn.preprocessing import *
```

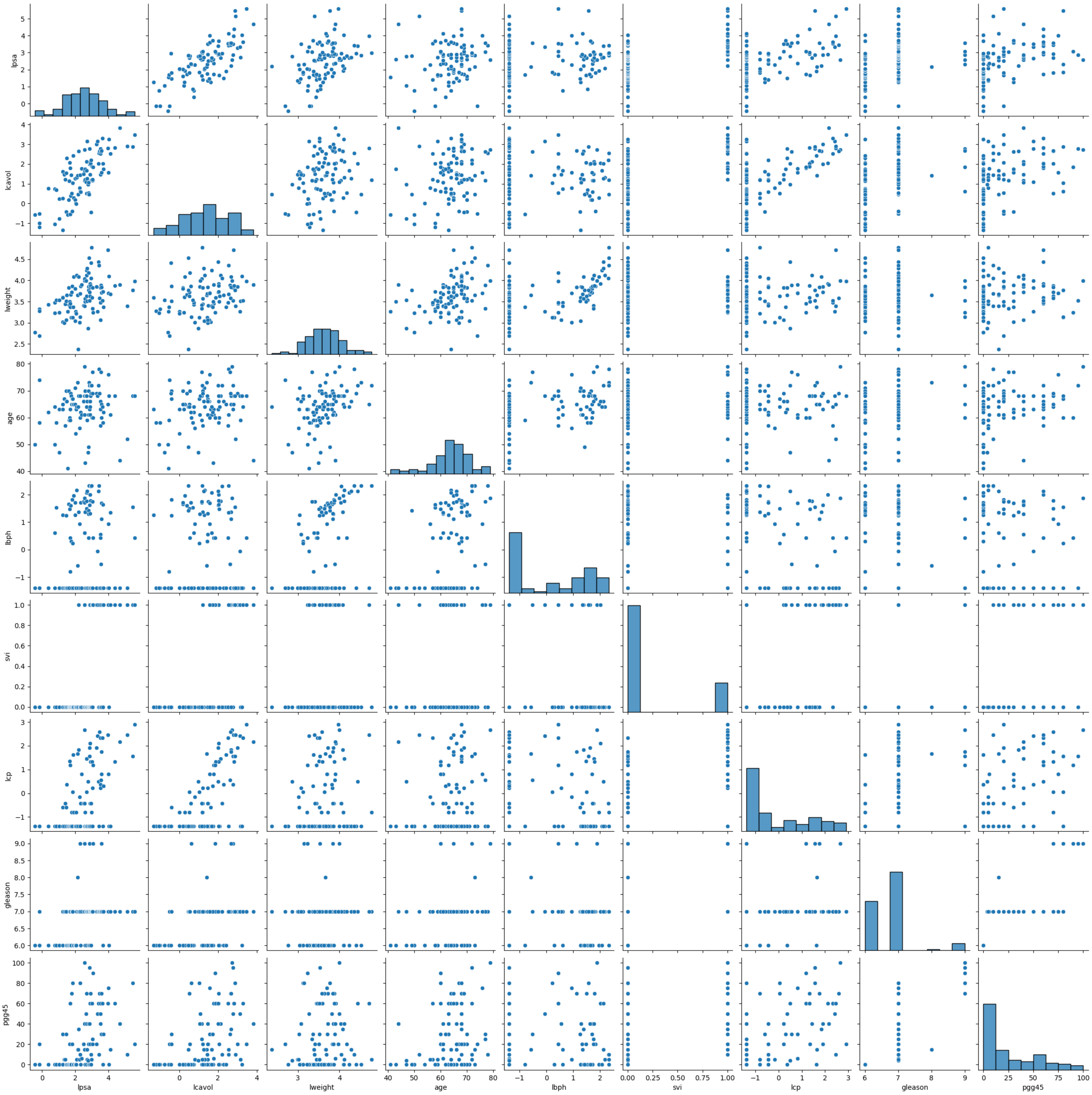
In [3]:
```
p = 67

pc_data = pd.read_csv("./hastie.su.domains_ElemStatLearn_datasets_prostate.data.csv")

train_df = pc_data.loc[pc_data.train == "T"]
test_df = pc_data.loc[pc_data.train != "T"]
```

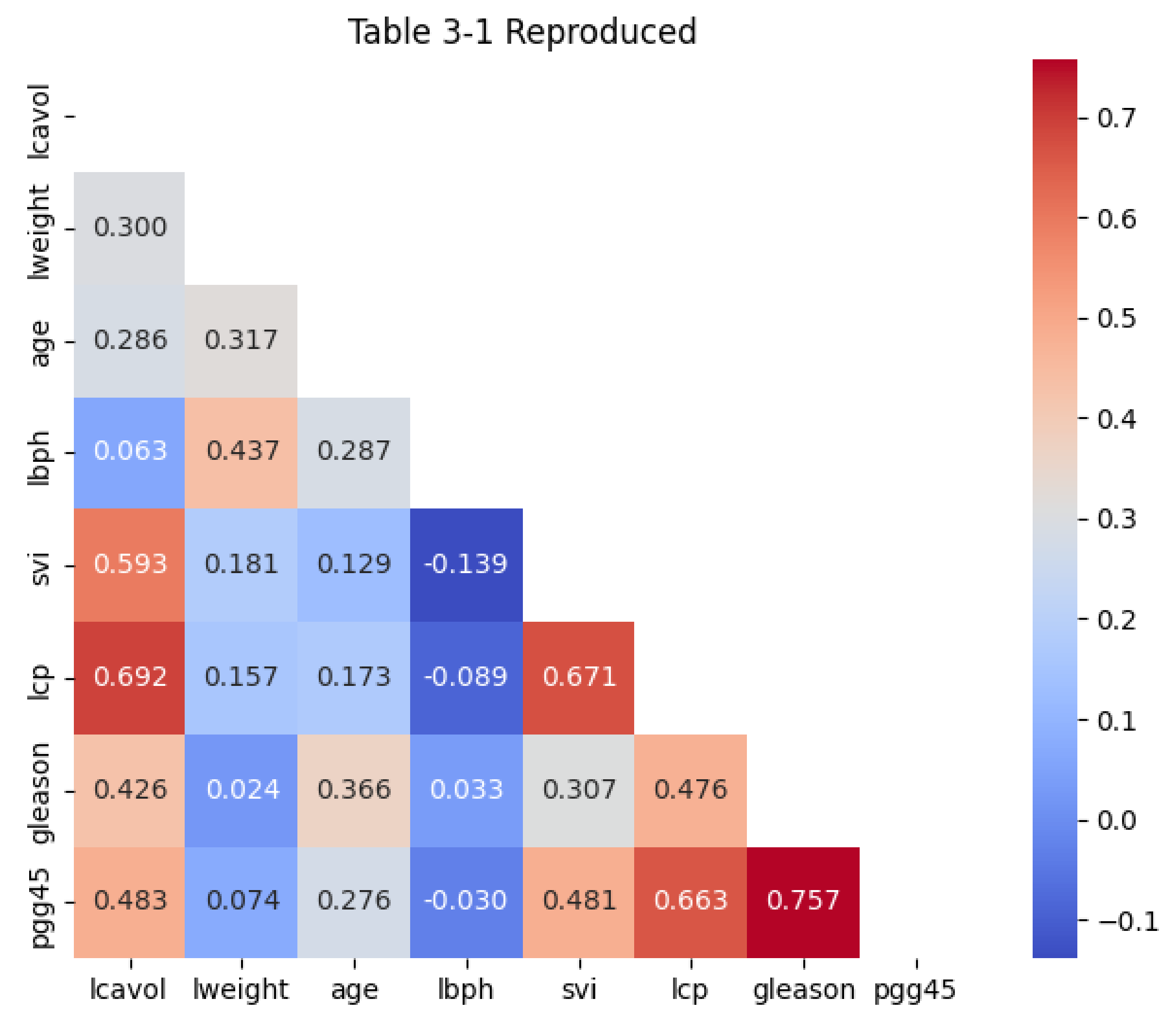
In [4]:
```
sns.pairplot(data = pc_data, vars = ["lpsa", "lcavol", "lweight", "age", "lbph", "svi", "lcp", "gleason", "pgg45"])
```

Out[4]: <seaborn.axisgrid.PairGrid at 0x272a9107bb0>



In [5]:
```python
correlation_matrix = train_df.drop(["lpsa", "train"], axis = 1).corr()

mask = np.triu(np.ones_like(correlation_matrix, dtype=bool))
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, fmt=".3f", cmap="coolwarm", mask=mask, square=True)
plt.title("Table 3-1 Reproduced")
plt.show()
```



Next, perform the Least Square Estimation and reproduce Table 3-2.

In [6]:
```python
X_origin = pc_data.drop(["lpsa", "train"], axis=1)
y = train_df["lpsa"]
y_test = test_df["lpsa"]

# Scale the whole dataset
scaler = StandardScaler()
X = scaler.fit_transform(X_origin)[pc_data.train == "T"]
X_test = scaler.fit_transform(X_origin)[pc_data.train != "T"]

model = sm.OLS(y, sm.add_constant(X)).fit()

coefficients = model.params
stderr = model.bse
z_scores = model.tvalues

ind_dict = {
    "const": "Intercept",
    "x1": "lcavol",
    "x2": "lweight",
    "x3": "age",
    "x4": "lbph",
    "x5": "svi",
    "x6": "lcp",
    "x7": "gleason",
    "x8": "pgg45"
}
results_df = pd.DataFrame({"Coefficient": coefficients, "Std. Error": stderr, "Z-Score": z_scores}).rename(index = ind_dict)

y_hat = model.predict(sm.add_constant(X_test))
ls_error_rate = np.mean((y_test - y_hat) ** 2)
ls_std_error = np.std((y_test - y_hat) ** 2, ddof = 1) / np.sqrt(y_test.size)

results_df
```

Out[6]:

	Coefficient	Std. Error	Z-Score
Intercept	2.464933	0.089315	27.598203
lcavol	0.676016	0.125975	5.366290
lweight	0.261694	0.095134	2.750789
age	-0.140734	0.100819	-1.395909
lbph	0.209061	0.101691	2.055846
svi	0.303623	0.122962	2.469255
lcp	-0.287002	0.153731	-1.866913
gleason	-0.021195	0.144497	-0.146681
pgg45	0.265576	0.152820	1.737840

Finally, perform the LASSO, Ridge and Elastic Net Regression and reproduce Table 3-3. The result may slightly different to the original table.

In [7]:
```python
lasso_model = Lasso(alpha = 0.209)

lasso_model.fit(X, y)

lasso_coefficients = lasso_model.coef_
lasso_coefficients = np.insert(lasso_coefficients, 0, lasso_model.intercept_)

y_hat = np.squeeze(lasso_model.predict(X_test))
lasso_error_rate = np.mean((y_test - y_hat) ** 2)
lasso_std_error = np.std((y_test - y_hat) ** 2, ddof = 1) / np.sqrt(y_test.size)
```

In [8]:
```python
ridge_model = Ridge(alpha = 24)

ridge_model.fit(X, y)

ridge_coefficients = ridge_model.coef_
ridge_coefficients = np.insert(ridge_coefficients, 0, ridge_model.intercept_)

y_hat = ridge_model.predict(X_test)
ridge_error_rate = np.mean((y_test - y_hat) ** 2)
ridge_std_error = np.std((y_test - y_hat) ** 2, ddof = 1) / np.sqrt(y_test.size)
```

In [9]:
```
elastic_net_model = ElasticNetCV(cv = 10)

elastic_net_model.fit(X, y)

elastic_net_coefficients = elastic_net_model.coef_
elastic_net_coefficients = np.insert(elastic_net_coefficients, 0, elastic_net_model.intercept_)

y_hat = elastic_net_model.predict(X_test)
en_error_rate = np.mean((y_test - y_hat) ** 2)
en_std_error = np.std((y_test - y_hat) ** 2, ddof = 1) / np.sqrt(y_test.size)
```

In [10]:
```
ecdf = pd.DataFrame({
    "OLS": results_df["Coefficient"],
    "LASSO": lasso_coefficients,
    "Ridge": ridge_coefficients,
    "EN": elastic_net_coefficients
})

ecdf.loc["Test Error"] = {
    "OLS": ls_error_rate,
    "LASSO": lasso_error_rate,
    "Ridge": ridge_error_rate,
    "EN": en_error_rate
}

ecdf.loc["Std. Error"] = {
    "OLS": ls_std_error,
    "LASSO": lasso_std_error,
    "Ridge": ridge_std_error,
    "EN": en_std_error
}
ecdf
```

Out[10]:

	OLS	LASSO	Ridge	EN
Intercept	2.464933	2.468346	2.464223	2.466754
lcavol	0.676016	0.535779	0.420106	0.657313
lweight	0.261694	0.187473	0.237861	0.260974
age	-0.140734	0.000000	-0.048296	-0.132089
lbph	0.209061	0.000000	0.161845	0.203565
svi	0.303623	0.085237	0.226399	0.294475
lcp	-0.287002	0.000000	-0.001086	-0.253085
gleason	-0.021195	0.000000	0.040716	-0.001772
pgg45	0.265576	0.006006	0.132123	0.236463
Test Error	0.521274	0.478962	0.490194	0.507957
Std. Error	0.178724	0.164466	0.162157	0.171556