Regularization

Regularization for prediction

In [1]:

```
# Load packages
import pandas as pd
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import Ridge
from sklearn.linear_model import Lasso
```

In [2]:

```
# Read data
df = pd.read_csv('rent.csv')
```

In [3]:

```
# Inspect data set
df.head()
```

Out[3]:

	area	rent
0	58	1380
1	72	1440
2	55	960
3	129	2160
4	78	1134

In [4]:

```
# Prepare data
poly = PolynomialFeatures(3, include_bias=False)
values = poly.fit_transform(df[['area']])
names = poly.get_feature_names(['area'])
X = pd.DataFrame(values, columns=names)
y = df['rent']
```

In [5]:

```
X.head()
```

Out[5]:

	area	area^2	area^3
0	58.0	3364.0	195112.0
1	72.0	5184.0	373248.0
2	55.0	3025.0	166375.0
3	129.0	16641.0	2146689.0
4	78.0	6084.0	474552.0

Ridge

In [6]:

```
# Fit model to data
model = Ridge(alpha=5)
model.fit(X, y)
```

Out[6]:

```
Ridge(alpha=5, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001)
```

In [7]:

```
# Display intercept
model.intercept_
```

Out[7]:

```
984.2201143848116
```

In [8]:

```
# Display betas
pd.Series(model.coef_, index=X.columns)
```

Out[8]:

```
area     -11.105354
area^2     0.324792
area^3    -0.001361
dtype: float64
```

In [9]:

```
# Make predictions
predictions = model.predict(X)
predictions[:5]
```

Out[9]:

```
array([1167.13552183, 1360.31466935, 1129.46207515, 2034.55863288,
       1448.10685116])
```

Lasso

In [10]:

```
# Fit model to data
model = Lasso(alpha=5, max_iter=100000)
model.fit(X, y)
```

Out[10]:

```
Lasso(alpha=5, copy_X=True, fit_intercept=True, max_iter=100000,
   normalize=False, positive=False, precompute=False, random_state=None,
   selection='cyclic', tol=0.0001, warm_start=False)
```

In [11]:

```
# Display betas
pd.Series(model.coef_, index=X.columns)
```

Out[11]:

```
area     -2.138984
area^2    0.215958
area^3   -0.000943
dtype: float64
```

In [12]:

```
# Make predictions
predictions = model.predict(X)
predictions[:5]
```

Out[12]:

```
array([1170.8727208 , 1365.97794644, 1131.1805645 , 2045.82907382,
       1451.97070537])
```

Regularization for classification

In [13]:

```
# Load packages
import pandas as pd
from sklearn.linear_model import LogisticRegression
```

In [14]:

```
# Load data
df = pd.read_csv('default.csv')
```

In [15]:

```
df.head()
```

Out[15]:

	default	student	balance	income
0	1	Yes	2135	20331
1	1	Yes	1475	20210
2	1	Yes	1102	17392
3	1	Yes	1789	17668
4	1	No	1891	34449

In [16]:

```
# Prepare data
predictors = ['student', 'balance', 'income']
X = pd.get_dummies(df[predictors], columns=['student'], drop_first=True)
y = df['default']
```

In [17]:

```
X.head()
```

Out[17]:

	balance	income	student_Yes
0	2135	20331	1
1	1475	20210	1
2	1102	17392	1
3	1789	17668	1
4	1891	34449	0

Ridge

In [18]:

```
# Fit model to data
model = LogisticRegression(C=10, tol=1e-6)
model.fit(X.values, y.values)
```

Out[18]:

```
LogisticRegression(C=10, class_weight=None, dual=False, fit_intercept=Tru
e,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=1e-06,
          verbose=0, warm_start=False)
```

In [19]:

```
# Display betas
pd.Series(model.coef_[0], index=X.columns)
```

Out[19]:

```
balance        0.006671
income        -0.000063
student_Yes   -0.752421
dtype: float64
```

Lasso

In [20]:

```
# Fit model to data
model = LogisticRegression(penalty='l1', C=10, tol=0.000001)
model.fit(X.values, y.values)
```

Out[20]:

```
LogisticRegression(C=10, class_weight=None, dual=False, fit_intercept=Tru
e,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l1', random_state=None, solver='liblinear', tol=1e-06,
          verbose=0, warm_start=False)
```

In [21]:

```
# Display betas
pd.Series(model.coef_[0], index=X.columns)
```

Out[21]:

```
balance        0.007946
income        -0.000031
student_Yes    0.000000
dtype: float64
```