Linear regression

Wojciech Wideł

Recap of the previous lecture

1. Select potential classes of models to be used (regression vs classification, predictive vs explanatory, parametric vs non-parametric).
2. Explore data in search of relationships.
3. Preprocess data (missing, outliers).
4. Feature engineering (non-linear transformations, encodings, scalings).
5. Model selection (cross-validation).
6. Analyse results produced by the best model. Are they satisfactory? Is there any bias? Does the model struggle with specific data? If yes, go back to step 4.

Reminder 1: hypothesis testing, simplified

- **Setup**: a sample $X_i \overset{\text{iid}}{\sim} F(\theta)$, null hypothesis $H_0\colon \theta \in \Theta_0$, assumed to be true, alternative hypothesis $H_1\colon \theta \in \Theta \setminus \Theta_0$
- **Step 1**: choose a statistic $f(X_1, \ldots, X_n)$ relevant to the hypotheses, such that the distribution of f is known under H_0
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $f(X_1, \ldots, X_n) \in I$
- **Step 3**: decide on *significance level* $\alpha = \max P(\text{reject } H_0 \text{ when } H_0 \text{ is true})$; use α to derive I
- **Step 4**: conclude, based on the form of the test and the set I derived in the previous step
- **Step 5**: compute *p-value*, the probability of obtaining test results at least as extreme as the result actually observed, under the assumption that the null hypothesis is true

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \ldots, X_n \stackrel{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \ldots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: for now, will use $f(X_1, \ldots, X_n) = \overline{X}$

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \ldots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: for now, will use $f(X_1, \ldots, X_n) = \overline{X}$
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $\overline{X} > c$, for some c yet to be determined

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \ldots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: for now, will use $f(X_1, \ldots, X_n) = \overline{X}$
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $\overline{X} > c$, for some c yet to be determined
- **Step 3**: set significance level α, usually 0.05. Then

$$\alpha = \max P(\text{ reject } H_0 \text{ when } H_0 \text{ is true }) = \max P(\overline{X} > c \text{ when } \mu = \mu_0)$$

$$= \max P\left(\frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}} > \frac{c - \mu_0}{\sigma/\sqrt{n}} \text{ when } \mu = \mu_0\right) \overset{H_0}{=} \max P\Big(Z > \underbrace{\frac{c - \mu_0}{\sigma/\sqrt{n}}}_{z_\alpha}\Big), Z \sim \mathcal{N}(0, 1).$$

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \ldots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: for now, will use $f(X_1, \ldots, X_n) = \overline{X}$
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $\overline{X} > c$, for some c yet to be determined
- **Step 3**: set significance level α, usually 0.05. Then

$$\alpha = \max P(\text{ reject } H_0 \text{ when } H_0 \text{ is true }) = \max P(\overline{X} > c \text{ when } \mu = \mu_0)$$

$$= \max P\left(\frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}} > \frac{c - \mu_0}{\sigma/\sqrt{n}} \text{ when } \mu = \mu_0\right) \overset{H_0}{=} \max P\Big(Z > \underbrace{\frac{c - \mu_0}{\sigma/\sqrt{n}}}_{z_\alpha}\Big), Z \sim \mathcal{N}(0, 1).$$

- **Step 4**: conclude: reject H_0 in favour of H_1 if $\overline{X} > \mu_0 + z_\alpha \sigma/\sqrt{n}$

Reminder 1: hypothesis testing, example

- **Setup**: $X_1, \dots, X_n \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: for now, will use $f(X_1, \dots, X_n) = \overline{X}$
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $\overline{X} > c$, for some c yet to be determined
- **Step 3**: set significance level α, usually 0.05. Then

$$\alpha = \max P(\text{ reject } H_0 \text{ when } H_0 \text{ is true }) = \max P(\overline{X} > c \text{ when } \mu = \mu_0)$$

$$= \max P\left(\frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}} > \frac{c - \mu_0}{\sigma/\sqrt{n}} \text{ when } \mu = \mu_0\right) \overset{H_0}{=} \max P\Big(Z > \underbrace{\frac{c - \mu_0}{\sigma/\sqrt{n}}}_{z_\alpha}\Big), Z \sim \mathcal{N}(0, 1).$$

- **Step 4**: conclude: reject H_0 in favour of H_1 if $\overline{X} > \mu_0 + z_\alpha \sigma/\sqrt{n}$
- **Step 5**: p-value here is $P(Z > (\overline{X} - \mu_0)/(\sigma/\sqrt{n}))$

Reminder 1: hypothesis testing, example reformulated

- **Setup**: $X_i \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$, σ^2 known, null hypothesis H_0: $\mu = \mu_0$, alternative hypothesis H_1: $\mu \neq \mu_0$, for some fixed μ_0
- **Step 1**: statistic of choice: **t-statistic** $f(X_1, \ldots, X_n) = \frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}}$.
  By the central limit theorem, $\overline{X} \sim \mathcal{N}(\mu, \sigma^2/n)$, and so, under the null hypothesis, $\frac{\overline{X} - \mu_0}{\sigma/\sqrt{n}} \sim \mathcal{N}(0, 1)$.
- **Step 2**: give the form of the test: reject H_0 in favour of H_1 if $f(X_1, \ldots, X_n) > z_\alpha$
- **Step 3**: set significance level α, compute corresponding z_α
- **Step 4**: conclude: reject H_0 in favour of H_1 if $f(X_1, \ldots, X_n) > z_\alpha$
- **Step 5**: p-value here is $P(Z > f(X_1, \ldots, X_n))$

Reminder 2: Bayes' theorem

$$p(\Theta|D) = \frac{p(D|\Theta)p(\Theta)}{p(D)}$$

component	**meaning**
D	data
Θ	model parameters
$p(D\|\Theta)$	data likelihood
$p(\Theta)$	prior parameters distribution
$p(D)$	data distribution, constant, irrelevant
$p(\Theta\|D)$	**posterior parameters distribution**

Reminder 3: linear regression so far

- Training data: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$, with $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip}) \in \mathbb{R}^p$, $y_i \in \mathbb{R}$
- Training data in matrix form: $X = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$, $y = (y_1, \ldots, y_n)^T$
- Assumed real relationship: $y_i = f(x_i) + \epsilon_i$, $\epsilon_i \overset{\text{iid}}{\sim} (0, \sigma^2)$
- Model: $\hat{y} = X\beta^T$, $(\beta_1, \ldots, \beta_p) \in \mathbb{R}^p$
  Note: this form assumes that either X contains a column of ones (with the corresponding coefficient being the intercept) or that both X and y have zero mean (in which case the intercept is known to be zero).
- Loss function: $L(\beta) = \frac{1}{n}||y - \hat{y}||^2$ (MSE)
- Regularization terms: $\alpha||\beta||_1$ (L1/lasso), $\alpha||\beta||_2^2$ (L2/ridge)

Optimal coefficients

Coefficients minimizing MSE can be determined by rewriting the loss function as

$$||\hat{y} - y||^2 = (X\beta^T - y)^T(X\beta^T - y) = \beta X^T X\beta^T - \beta X^T y - y^T X\beta^T + y^T y$$

and solving

$$\frac{\partial L}{\partial \beta} = 2X^T X\beta - 2X^T y = 0.$$

This yields the **ordinary least squares estimates** of the coefficients

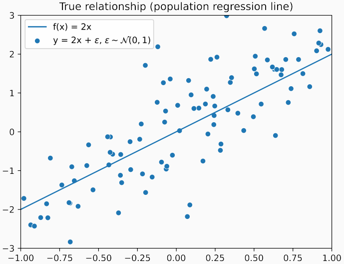
$$\widehat{\beta}_{OLS} = (X^T X)^{-1} X^T y.$$

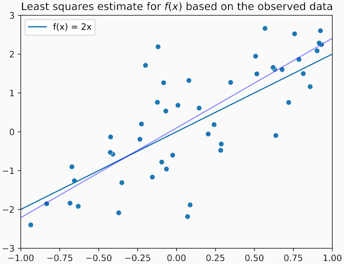
Similarly, the coefficients minimizing **MSE loss under ridge regularization** are

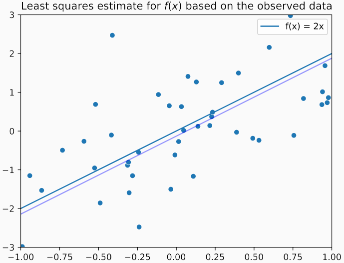
$$\widehat{\beta}_{RR} = (X^T X + \alpha \mathbb{I})^{-1} X^T y.$$

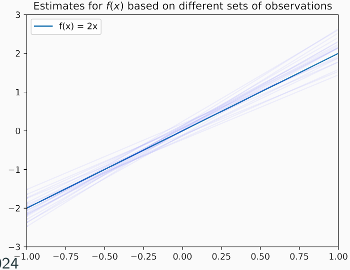
No closed-form solution exists for lasso regularization, because in this case the loss function is not differentiable. Nevertheless, we will use $\widehat{\beta}_{LR}$ to denote this solution.

Linear regression and hypothesis testing









Linear regression and hypothesis testing

sample mean	**OLS coefficients**
$X_i \overset{\text{iid}}{\sim} \mathcal{N}(\mu, \sigma^2)$	$y_i \overset{\text{iid}}{\sim} \mathcal{N}(\beta X_i, \sigma^2)$
μ, $\widehat{\mu} = \overline{X} \sim \mathcal{N}(\mu, \sigma^2/n)$	β, $\widehat{\beta}_{OLS} \sim \mathcal{N}(\beta, \sigma^2(X^TX)^{-1})$
$SE(\widehat{\mu})^2 = \sigma^2/n$	$SE(\widehat{\beta}_{OLS})$ derived from $\widehat{\beta}_{OLS}$
Is μ equal to zero? Compute the t-statistic $\widehat{\mu}/SE(\widehat{\mu})$ and the p-value	Is β equal to zero? Compute the t-statistic $\widehat{\beta}_{OLS}/SE(\widehat{\beta}_{OLS})$ and the p-value

Linear regression and hypothesis testing

```
import numpy as np
import statsmodels.api as sm

rng = np.random.Generator(np.random.PCG64(seed=72346))

x1 = np.linspace(-1, 1, 100)
x2 = x1**2
x3 = rng.standard_normal(100)
epsilon = rng.standard_normal(100) / 2

X = np.matrix([x1, x2, x3]).T
y = x1 + 2 * x2 + epsilon #no dependence on x3
est = sm.OLS(y, X)
est2 = est.fit()
print(est2.summary())
```

Linear regression and hypothesis testing

```
'''
                            OLS Regression Results
==============================================================================
(...)
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.9352      0.089     10.461      0.000       0.758       1.113
x2             2.1422      0.114     18.749      0.000       1.915       2.369
x3            -0.0157      0.054     -0.288      0.774      -0.124       0.092
==============================================================================
Omnibus:                        0.873   Durbin-Watson:                   1.754
Prob(Omnibus):                  0.646   Jarque-Bera (JB):                0.733
Skew:                           0.209   Prob(JB):                        0.693
Kurtosis:                       2.968   Cond. No.                         2.12
==============================================================================
(...)
'''
```

Bayesian linear regression

- Assumed real relationship, rewritten: $y|\beta, \sigma \sim \mathcal{N}(X\beta^T, \sigma^2\mathbb{I})$
- Errors assumed to be independent (ordinary regression)

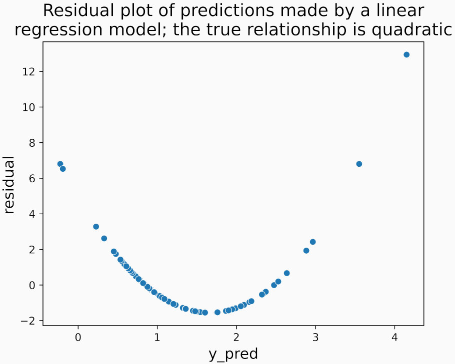
priors	**posteriors**
$p(\beta\|\sigma) \propto 1$ (uniform) $p(\sigma^2) \propto \frac{1}{\sigma^2}$ (e.g., inverse uniform)	$\beta\|X, y \sim t_{p+1}$, centered at $\widehat{\beta}_{OLS}$ $\sigma^2\|X, y \sim IG(\cdot, \cdot)$
$\beta \sim \mathcal{N}(\beta^0, \sigma^2\Sigma)$ $\sigma^2 \sim IG(a_0, b_0)$	$\beta\|\sigma, y \sim \mathcal{N}(\cdot, \cdot)$ $\sigma^2\|y \sim IG(\cdot, \cdot)$
$\beta_i \overset{\text{iid}}{\sim} \mathcal{L}(0, b)$	$\widehat{\beta}_{LR}$ for $\alpha = 2\sigma^2/b$ is the mode of β's posterior distribution,
$\beta_i \overset{\text{iid}}{\sim} \mathcal{N}(0, c)$	$\widehat{\beta}_{RR}$ for $\alpha = 2\sigma^2/c$ is both the mean and the mode of β's posterior distribution

Potential problems, 1/3

Correlation of error terms: estimated standard errors tend to underestimate the true standard errors, resulting in narrower confidence and prediction intervals; may lead to unwarranted confidence in the model

Potential problems, 2/3

Non-constant variance of error terms: can be identified in residual plots ($\widehat{y}$ vs $y - \widehat{y}$); suggests that the model is biased. One possible solution is to model a non-linear transformation of y instead of y itself.



Potential problems, 3/3

Multicollinearity: it can be difficult to separate out the individual effects of strongly correlated variables on the response; compute *variance inflation factors* to identify problematic variables:

$$\mathsf{VIF}(X_i) = 1/(1 - R^2_{X_i|X_{-i}}),$$

where $R^2_{X_i|X_{-i}}$ denotes the R^2 from a regression of X_i on all the other predictors. Values exceeding five (corresponding to $R^2_{X_i|X_{-i}}$ greater than 0.8) indicate multicollinearity.

Solutions: drop or combine (e.g., using their average after standardization) the problematic variables.

Variance inflation factors in Python

```
import numpy as np
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.tools import add_constant

rng = np.random.Generator(np.random.PCG64(seed=72346))
x1 = rng.standard_normal(100)
x2 = rng.standard_normal(100)
x3 = rng.standard_normal(100)
x4 = x1 + 3 * x2 + rng.standard_normal(100)
df = pd.DataFrame({f"x{i+1}": [x1, x2, x3, x4][i] for i in range(4)})
X = add_constant(df)  # expected by variance_inflation_factor
pd.Series(
[variance_inflation_factor(X.values, i) for i in range(X.shape[1])], index=X.columns)
#const     1.008653
#x1        1.461295
#x2        7.311439
#x3        1.022520
#x4        7.745740
```

Model selection/hyperparameters tuning via grid search

```python
from sklearn.preprocessing import StandardScaler, MinMaxScaler
from sklearn.linear_model import Lasso, Ridge
from sklearn.pipeline import Pipeline
from sklearn.model_selection import cross_val_score

X = df[features]
y = df[target]
for scaler in [StandardScaler, MinMaxScaler]:
    for m in [Lasso, Ridge]:
      for alpha in np.linspace(0, 30, 1000):
        model = m(alpha=alpha)
        pipeline = Pipeline([("scaler", scaler(), ("model", model)])
        cv = KFold(n_splits=10)
        scores = cross_val_score(pipeline, X, y, cv=cv, scoring="neg_mean_squared_error")
        score = -np.mean(scores)
        # code for keeping track of scores omitted here
```

Model selection/hyperparameters tuning using hyperopt-sklearn

```
from hpsklearn import HyperoptEstimator, linear_regression, lasso, ridge , any_preprocessing
from hyperopt import tpe
from hyperopt import hp
from sklearn.metrics import mean_squared_error

X_train = df[features]
y_train = df[target]
reg_alpha = hp.loguniform("alpha", low=np.log(1e-5), high=np.log(50))
models = hp.choice("regressor",
        [linear_regression("lr"),
        lasso("lasso", alpha=reg_alpha),
        ridge("ridge", alpha=reg_alpha)])
estim = HyperoptEstimator(regressor=models, preprocessing=any_preprocessing("my_pre"),
            algo=tpe.suggest, max_evals=200,
            trial_timeout=120, loss_fn=mean_squared_error)
estim.fit(X_train, y_train, n_folds=5, cv_shuffle=True)
print(estim.best_model())
```

Summary

1. P-values in ordinary least squares regression allow for assessing features importance
2. Coefficients of OLS, lasso and ridge regression lines estimated by minimizing MSE correspond to coefficients in Bayesian linear regression under appropriate priors
3. Use variance inflation factors for assessing multicollinearity of predictors
4. Use residual plots for checking model bias
5. **Use hyperopt-sklearn for model selection!**

References

- G. James, D. Witten, T. Hastie and R. Tibshirani, *An Introduction to Statistical Learning*
- https://ekamperi.github.io/mathematics/2020/08/02/bayesian-connection-to-lasso-and-ridge-regression.html
- https://de.mathworks.com/help/econ/what-is-bayesian-linear-regression.html
- https://statswithr.github.io/book/introduction-to-bayesian-regression.html#sec:Bayes-multiple-regression
- https://gregorygundersen.com/blog/2021/08/26/ols-estimator-sampling-distribution/
- hyperopt-sklearn paper: https://www.automl.org/wp-content/uploads/2019/05/AutoML_Book_Chapter5.pdf