Computational Statistics and Data Analysis (MVComp2)

Exercise 10

Lecturer Tristan Bereau

Semester Wi23/24
Due Jan. 18, 2024, 23:59

1 Log-likelihood for multinomial logistic regression (3 points)

Recall binary logistic regression. We consider multinomial logistic regression—an extension to C classes. We write the classifier as a categorical distribution, Cat, a discrete probability distribution. For a datapoint $\boldsymbol{x}_n \in \mathbb{R}^D$, we have the discriminative classification model

$$p(y_n|\boldsymbol{x}_n, \boldsymbol{\theta}) = \mathrm{Cat}(y_n|\mathrm{softmax}(\mathbf{W}^\top \boldsymbol{x}_n)) = \prod_{c=1}^{C} \mathrm{softmax}(\boldsymbol{w}_c^\top \boldsymbol{x}_n)^{\mathbb{1}(y_n=c)},$$

where $\mathbf{W}$ is a $C \times D$ weight matrix for C classes and D features, $\mathbb{1}(\cdot)$ is the indicator function,[1] and the softmax function is defined below. We can rewrite the class label as a one-hot encoding: $y_{nc} = \mathbb{1}(y_n = c)$. This yields

$$p(y_{nc} = 1|\boldsymbol{x}_n, \boldsymbol{\theta}) = \mu_{nc} = \mathrm{softmax}(\boldsymbol{\eta}_{nc}) = \frac{\mathrm{e}^{\eta_{nc}}}{\sum_{c'=1}^{C} \mathrm{e}^{\eta_{nc'}}},$$

where $\boldsymbol{\eta}_{nc} = \boldsymbol{w}_c^\top \boldsymbol{x}_n$ is the vector of logits for the n-th datapoint and class c.

(a) Show that the Jacobian of the softmax is

$$\frac{\partial \mu_{ik}}{\partial \eta_{ij}} = \mu_{ik}(\delta_{kj} - \mu_{ij}),$$

where δ_{kj} is the Kronecker delta.

(b) Show that the gradient of the negative log-likelihood is given by

$$\nabla_{\boldsymbol{w}_c} \ell = \sum_i (\mu_{ic} - y_{ic}) \boldsymbol{x}_i.$$

(c) Show that the Hessian block between classes c and c' is given by

$$\mathbf{H}_{c,c'} = \mu_{nc}(\delta_{c'c} - \mu_{nc'}) \boldsymbol{x}_n \boldsymbol{x}_n^\top.$$

[1] $\mathbb{1}(\cdot)$ is 1 if its argument is satisfied and 0 otherwise.

2 Classification of penguins (4 points)

Download the following dataset about penguins: penguins.csv. We will focus on the following features:

Variable	Description
`species`	Penguin species
`bill_length_mm`	bill (or beak) length in mm
`bill_depth_mm`	bill depth in mm
`flipper_length_mm`	flipper (or wing) length in mm

The objective of the exercise is to construct a classifier for the response variable `species` from the other features. The dataset contains three penguin species: Adelie, Chinstrap, and Gentoo.

(a) For each species, plot the one-dimensional cumulative distribution functions of the different features. Argue whether one-dimensional classifiers are likely to perform well to seperate each species.

(b) Let's classify penguins. To this end, assign a (uninformative) uniform prior distribution for each species. Model the likelihood using univariate Gaussian distributions. Write a function that can compute the posterior probability of each species given a feature and its value. Make sure to normalize your probabilities. Test your model for a flipper length of 213 mm, as well as 197 mm. Comment on the results.

(c) Use the one-dimensional posterior distributions from (b) applied to the dataset to evaluate their performance as classifier. Classify according to the highest probability encountered. Evaluate the resulting proportion of correctly predicted labels for each feature across your dataset.

(d) Use the prior and likelihood distributions of part (b) to build a Naïve Bayes classifier across the three features. Evaluate the performance of the classification by measuring the proportion of correctly predicted labels. Compare to (c) and comment.

Hint: You may find the following functions useful:

- `empiricaldist.Cdf`
- `empiricaldist.Pmf`
- `scipy.stats.norm`

3 Prostate cancer, kernelized (3 points)

Let's revisit the prostate-cancer dataset from Homework Set 8, which you can download here: prostate.csv. This time we will use kernel ridge regression to build a supervised learning model for `lpsa`.

(a) Implement yourself a kernel ridge regression. Do not use existing statistics / machine learning libraries (though feel free to use linear-algebra libraries). To simplify your task, consider extending the following template that inherits from `OrdinaryLeastSquares` from Homework Set 8 (and provided below):

```python
import numpy as np
from dataclasses import dataclass, field

@dataclass
class OrdinaryLeastSquares:
    model_params: np.ndarray = field(init=False)
    training_set: np.array = field(init=False)

    def __post_init__(self):
        self.model_params = None

    def fit(self, X_train_: np.ndarray, y_train_: np.ndarray) -> None:
        self.training_set = X_train_
        self.model_params = (
            np.linalg.inv(X_train_.T @ X_train_) @ X_train_.T @ y_train_
        )

    def predict(self, X_test_: np.ndarray) -> np.ndarray:
        return X_test_ @ self.model_params

    def rmse(self, X_test_: np.ndarray, y_test_: np.ndarray) -> float:
        y_pred = self.predict(X_test_)
        return np.sqrt(mean_squared_error(y_test_, y_pred))

@dataclass
class GaussianKernel(OrdinaryLeastSquares):
    sigma: float
    regularization: float

    def kernel_matrix(self, x_1: np.array, x_2: np.array) -> np.array:
        pass

    def fit(self, x: np.array, y: np.array) -> None:
        pass

    def predict(self, x: np.array) -> np.array:
        pass
```

where `sigma` is the length scale of a Gaussian (i.e., “radial basis function”) kernel and `regularization` is the coefficient in front of the ℓ_2 parameter. The function `kernel_matrix` computes the kernel matrix between any two datasets $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$. Use the same train/test split as in Homework set 8. Generate a contour plot of the *test* root-mean-squared error (RMSE) as a function of σ and λ with suggested ranges $10^{-1} \leq \sigma \leq 10^4$ and $10^{-6} \leq \lambda \leq 10^4$. Interpret the results: What happens at low and high σ and λ, and why?

Hint: You may find the following functions useful:

- `numpy.logspace`
- `numpy.meshgrid`
- `matplotlib.pyplot.contourf`

(b) Use part (a) to identify optimal hyperparameters for σ and λ. Generate a parity plot of predicted against reference labels for the test dataset. Compare the value of the RMSE to your results from ordinary least squares and linear ridge regression in Homework Set 8.