MADS-ML – Machine Learning

Regression

Prof. Dr. Stephan Doerfel

FACHHOCHSCHULE KIEL
University of Applied Sciences

Moodle (WiSe 2024/25)

Regression

- supervised task
- predict continuous values,e.g. prices, time windows
- given: independent (or explanatory) variables, called predictors
- target: dependent variable (result)

Examples: stock market prediction, rating prediction

Regression vs. Classification

- different target: continuous vs. categorical
- → different evaluation measures: instead of counting correct vs. incorrect, we must evaluate close vs. far-off predictions
- → different optimization goal
- same train-test setup: splitting, cross validation
- no simple equivalent for stratification (definition requires classes)

Outline

Evaluation – Mean Squared Error

- MSE – the most popular evaluation metric:

$$\text{MSE} = \frac{1}{|D|} \sum_{d=1}^{|D|} (y_d - \hat{y}_d)^2$$

- compute the **residuals** for all data instances $(y_d - \hat{y}_d)$
- large residuals are punished over-proportionally harder than small ones
- MSE of 0 means perfect fit
- smooth function (no absolute value necessary due to squaring)
- caveat: interpretability
- in Python: `sklearn.metrics.mean_squared_error`

Coefficient of Determination

- R^2 – standardized version of MSE:

$$R^2 = 1 - \frac{\sum_{d=1}^{|D|} (y_d - \hat{y}_d)^2}{\sum_{d=1}^{|D|} (y_d - \bar{y})^2} = 1 - \frac{\text{MSE}}{\text{var}(y)}$$

- rescaled (and inverse) version of MSE
- $\bar{y}$ is the mean of the values y_d
- $R^2 \leq 1$ (1 means perfect predictions)
- interpretation: share of the target's variance that is explained by the estimator (by the predictions $\hat{y}$)

Mean Average Error

- MAE:

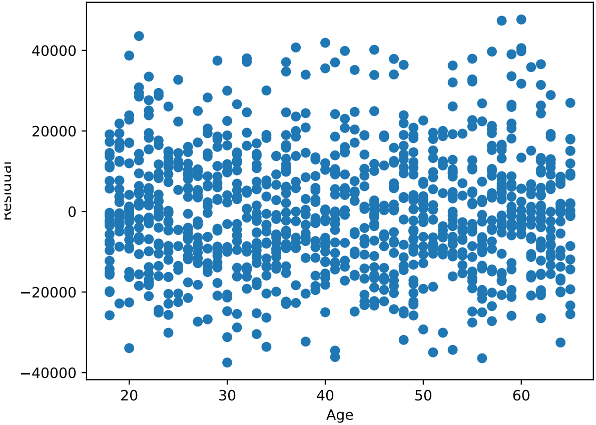
$$\text{MAE} = \frac{1}{|D|} \sum_{d=1}^{|D|} |y_d - \hat{y}_d|$$

- MAE of 0 means perfect fit
- interpretability: the sum of the residuals
- caveat: absolute value adds complexity in derivation (problematic for gradient descent)
- in Python: `sklearn.metrics.mean_squared_error` with `squared=False`

Mean Average Percentage Error

- MAPE:
$$\frac{1}{|D|}\sum_{d=1}^{|D|}\left|\frac{y_d-\hat{y}_d}{y_d}\right|$$
- interpretation: the residuals as share of the actual value
- MAPE of 0 means perfect fit
- absolute interpretation possible (whereas MSE and MAE depend on the context of the actual values)
- caveat: derivation analogous to MAE
- caveat: only possible if the original value is non-zero

Analysis of the Residual Plot



- show the residuals (errors) in one plot
- the residuals should be close to zero
- the residuals should be distributed at random (no systematic

Outline

Correlation

- correlation measures the degree of association between two variables
- correlation can be expressed by various different measures
  - Pearson's r for linear correlation
  - Spearman's rank correlation ρ (Pearson's r on ranks)
  - Kendall's τ for rank correlation (concordant pairs)
- can be used as distance functions between two series of values

Correlation – Pearson's r

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

- r is a measure of linear correlation between two variables
- linearly correlated variables grow or decrease proportionally
- $-1 \leq r \leq 1$ where
  - -1 means anti-correlation
  - 0 means no correlation
  - 1 means correlation

Correlation – Caveat!

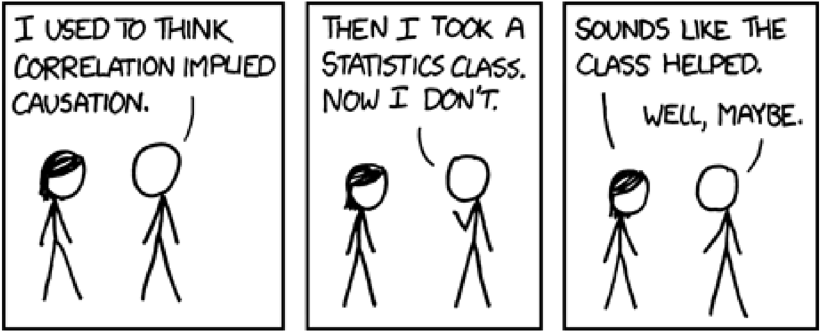
- when testing for correlation consider not only the value of r but also the **p-value** – a measure for the probability of receiving r or a more extreme result for uncorrelated (independent) variables
- even correlation with low p-value does NOT mean that two variables are actually related – spurious correlations
- Finding correlations with low p-values is a random experiment – the more candidates are tested, the more likely it is that one of them will be significant (low p-value) → **Bonferroni correction**

Correlation and Causation

What can we learn from correlation (between A and B about causation?

- Correlation can occur due to
  - causal relations: A leads to B, or B leads to A, e.g. working hours and pay
  - common causes (hidden variable): C leads to A and to B, e.g. ice cream sales, violent crime rates, temperature
  - Coincidence!
- Even when a causality seems plausible at first glance . . . !

Correlation and Causation



Source: XKCD

Outline

Linear Regression

- goal: Determine a linear relationship between the explanatory variables and the target
- model: $\hat{y} = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + w_0$ where
  - x is the vector of explanatory variables (features),
  - w is a vector of weights (one per feature), and
  - w_0 is the intercept of the model
- learning: gradient descent
- works well for linear correlated features

The Model

For a dataset D of d-dimensional vectors $\boldsymbol{x}$:

$$\hat{y} := \sum_{i=1}^{d} (w_i x_i) + w_0$$

- w_i are the coefficients of the respective features, called **population parameters**
- $e := y - \hat{y}$ is called the **residual** of the regression, an error term modeling
  - measurement errors
  - influence of other features
  - non-linear effects
  - otherwise unexplained influences

Best Fit

- Minimize a loss function to find the best linear model fitting the data
- Loss function: sum of the squared errors (residuals) ➔ MSE
- (Alternative: MAE – less sensitive to outliers, computationally more complex)
- Initialize parameters at random, compute the loss and use gradient descent to update the parameters

Sidetrack: Gradient Descent 1/2

Given a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ that is differentiable in a neighborhood of $\boldsymbol{x} \in \mathbb{R}^n$.

- $\nabla f(\boldsymbol{x})$ is the vector of the partial derivatives of f at $\boldsymbol{x}$
- $f(\boldsymbol{x})$ decreases fastest from $\boldsymbol{x}$ in the direction of the negative gradient $-\nabla f(\boldsymbol{x})$ of f at $\boldsymbol{x}$
- thus, there exists a (potentially very small) $\alpha \in \mathbb{R}$ s.t. for $\boldsymbol{y} := \boldsymbol{x} - \alpha \cdot \nabla f(\boldsymbol{x})$ holds $f(\boldsymbol{x}) > f(\boldsymbol{y})$

General Idea: If we make small enough steps, starting from $\boldsymbol{x}$ we will get closer to (approximately reach) a local minimum of f – provided, one exists.

Sidetrack: Gradient Descent 2/2

In many machine learning algorithms, gradient descent is the mechanism for learning values.

- define a function to be minimized e.g. a loss function, a distance score, ...
- ensure that the function is differentiable in the relevant areas
- ensure that the function has local minima
- select the **learning rate** α
- start at some random point
- make steps of $-\alpha \cdot \nabla f(\boldsymbol{x_i})$ to find a point $\boldsymbol{x_j}$ until some criterion has been met
- accept the final $\boldsymbol{x_j}$ as approximation of the minimum

Notebook 09_1_gradient_descent_example

Gradient Descent for Linear Regression

$$f : \mathbb{R}^n \to \mathbb{R}$$

is the function that maps a particular choice for the population parameters to the resulting loss (MSE)

$$f : w_0, w_1, \ldots, w_d \mapsto \sum_{i \in D} (y^i - \hat{y}^i)^2$$

$$= \sum_{i \in D} (y^i - (w_0 + w_1 x_1 + \ldots + w_d x_d))^2$$

Outline

Interpretation of the Parameters

If the model describes the data well, then

- the intercept describes the (theoretical) base quantity if all other factors are zero
- the other parameters each describe the increase of the target if the corresponding feature is raised 1

This interpretation is a **ceteris paribus** argument, meaning one feature changes and all else is unchanged (all else equal).

Notebook 09_2_linear_regression_wages, Cells 1–21

Outline

Polynomial Regression

- Use the idea of Linear Regression
- Use polynomial features → compute products of monomials such that their degree (sum of the degrees of all factors in a monomial) does not exceed a threshold (hyperparameter!)
- example: features x_1, x_2, x_3, degree threshold 3:
  - degree 0: 1
  - degree 1: x_1, x_2, x_3
  - degree 2: $x_1^2, x_2^2, x_3^2, x_1x_2, x_1x_3, x_2x_3$
  - degree 3: $x_1^3, x_2^3, x_3^3, x_1^2x_2, x_1^2x_3, x_2^2x_1, x_2^2x_3, x_3^2x_1, x_3^2x_2, x_1x_2x_3$
- Run linear regression on these polynomial features.

Notebook 09_2_linear_regression_wages, Cells 22–28

Other Regressors

- kNN: sklearn.neighbors.KNeighborsRegressor
- Decision Trees: `sklearn.tree.DecisionTreeRegressor`
- SVMs: `sklearn.svm.SVR`
- Voting: `sklearn.ensemble.VotingRegressor`