Solution 1: Car Price Prediction

a) We face a **supervised regression** task: we definitely need labeled training data to infer a relationship between cars' attributes and their prices, and price in EUR is a continuous target (or quasi-continuous, to be exact – as with all other quantities, we can only measure it with finite precision, but the scale is sufficiently fine-grained to assume continuity). **Prediction** is definitely the goal here, however, it might also be interesting to examine the explanatory contribution of each feature.

b) Target variable and potential features:

Variable	**Role**	**Data type**
Price in EUR	Target	Numeric
Age in days	Feature	Numeric
Mileage in km	Feature	Numeric
Brand	Feature	Categorical
Accident-free y/n	Feature	Binary
...	...	...

c) Let x_1 and x_2 measure age and mileage, respectively. Both features and target are numeric and (quasi-) continuous. It is also reasonable to assume non-negativity for the features, such that we obtain $\mathcal{X} = (\mathbb{R}_0^+)^2$, with $\mathbf{x}^{(i)} = (x_1^{(i)}, x_2^{(i)})^\top \in \mathcal{X}$ for $i = 1, 2, \dots, n$ observations. As the standard LM does not impose any restrictions on the target, we have $\mathcal{Y} = \mathbb{R}$, though we would probably discard negative predictions in practice.

d) We can write the hypothesis space as:

$$\mathcal{H} = \{f(\mathbf{x} \mid \boldsymbol{\theta}) = \boldsymbol{\theta}^\top \mathbf{x} \mid \boldsymbol{\theta} \in \mathbb{R}^3\} = \{f(\mathbf{x} \mid \boldsymbol{\theta}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \mid (\theta_0, \theta_1, \theta_2) \in \mathbb{R}^3\}.$$

Note the **slight abuse of notation** here: in the lecture, we first define $\boldsymbol{\theta}$ to only consist of the feature coefficients, with $\mathbf{x}$ likewise being the plain feature vector. For the sake of simplicity, however, it is more convenient to append the intercept coefficient to the vector of feature coefficients. This does not change our model formulation, but we have to keep in mind that it implicitly entails adding an element 1 at the first position of each feature vector.

e) The parameter space is included in the definition of the hypothesis space and in this case given by $\Theta = \mathbb{R}^3$.

f) Loss function for the i-th observation: $L\left(y^{(i)}, f\left(\mathbf{x}^{(i)} \mid \boldsymbol{\theta}\right)\right) = \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)}\right)^2$.

g) In order to find the optimal $\hat{\boldsymbol{\theta}}$, we need to solve the following minimization problem:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \Theta} \mathcal{R}_{\text{emp}}(\boldsymbol{\theta}) = \arg\min_{\boldsymbol{\theta} \in \Theta} \left(\sum_{i=1}^n \left(y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} \right)^2 \right)$$

This is achieved in the usual manner of setting the derivative w.r.t. $\boldsymbol{\theta}$ to 0 and solving for $\boldsymbol{\theta}$, yielding the familiar least-squares estimator.

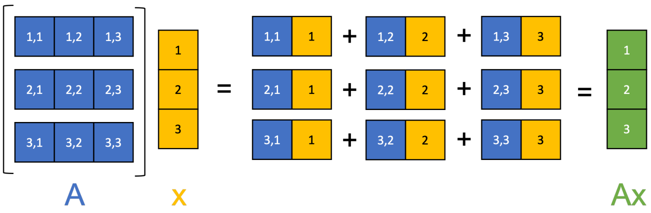
Solution 2: Vector Calculus

a) In computing $\mathbf{A}\mathbf{x}$ we multiply each of the m rows in $\mathbf{A}$ with the sole length-n column in $\mathbf{x}$, leaving us with a column vector $f(\mathbf{x}) \in \mathbb{R}^{m \times 1}$. Thus, we have $f : \mathbb{R}^{n(\times 1)} \to \mathbb{R}^{m(\times 1)}$.

The i-th function component $f_i(\mathbf{x})$ corresponds to multiplying the i-th row of $\mathbf{A}$ with $\mathbf{x}$, amounting to

$$f_i(\mathbf{x}) = \sum_{j=1}^{n} a_{ij} x_j,$$

with a_{ij} the element in the i-row, j-th column of $\mathbf{A}$.



https://mbernste.github.io/posts/matrix_vector_mult/

b)

i) The gradient is the row vector[1] of partial derivatives, i.e., the derivatives of f w.r.t. each dimension of $\mathbf{x}$:

$$\frac{\mathrm{d}f(\mathbf{x})}{\mathrm{d}\mathbf{x}} = \begin{pmatrix} \frac{\partial f(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f(\mathbf{x})}{\partial x_n} \end{pmatrix}.$$

Now, since f is a vector-valued function, each component is itself a vector of length m. Therefore, we have $\frac{\mathrm{d}f(\mathbf{x})}{\mathrm{d}\mathbf{x}} \in \mathbb{R}^{m \times n}$, given by the collection of all partial derivatives of each function component:

$$\frac{\mathrm{d}f(\mathbf{x})}{\mathrm{d}\mathbf{x}} = \begin{pmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_m(\mathbf{x})}{\partial x_n} \end{pmatrix}$$

This matrix is also called the *Jacobian* of f.

ii) We have

$$\frac{\partial f_i(\mathbf{x})}{\partial x_j} = \frac{\partial \left(\sum_{j=1}^{n} a_{ij} x_j \right)}{\partial x_j} = a_{ij}.$$

Doing this for every element yields

$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix},$$

and we have $\frac{\mathrm{d}f(\mathbf{x})}{\mathrm{d}\mathbf{x}} = \frac{\mathrm{d}\mathbf{A}\mathbf{x}}{\mathrm{d}\mathbf{x}} = \mathbf{A}$.

For more explanations and exercises, including a useful collection of rules for calculus, we recommend the book ”Mathematics for Machine Learning” (`https://mml-book.github.io/book/mml-book.pdf`).

[1] Pertaining to one of two conventions; we use the *numerator layout* here (the transposed version is called *denominator layout*).