

**Fakulta matematiky, fyziky a informatiky Univerzity Komenského,  
Bratislava**

**Projekt z lineárneho programovania  
A04 - Predikcia kvality vín**

*Piati proti optimalizácii*

Tomáš Antal, 2DAV, 0.2

Erik Božík, 2DAV, 0.2

Róbert Kendereš, 2DAV, 0.2

Teo Pazera, 2DAV, 0.2

Andrej Špitalský, 2DAV, 0.2

# Obsah

<b>0</b>	<b>Úvod</b>	<b>2</b>
<b>1</b>	<b>Úloha A</b>	<b>3</b>
1.1	Minimalizovanie $L^1$ normy . . . . .	3
1.1.1	Prípustnosť a optimalita . . . . .	3
1.2	Minimalizovanie $L^\infty$ normy . . . . .	4
1.2.1	Prípustnosť a optimalita . . . . .	5
<b>2</b>	<b>Úloha B</b>	<b>6</b>
2.1	Prevedenie úlohy LP do tvaru pre <code>scipy.optimize.linprog</code> . . . . .	6
2.2	Implementovanie regresných LP úloh . . . . .	6
<b>3</b>	<b>Úloha C</b>	<b>8</b>
<b>4</b>	<b>Úloha D</b>	<b>10</b>
<b>5</b>	<b>Úloha E</b>	<b>11</b>
5.1	Spracovanie všeobecnej triedy pre $L^1$ a $L^\infty$ regresiu . . . . .	11
5.2	Porovnanie použitia $L^1$ a $L^\infty$ lineárnej regresie . . . . .	12
5.2.1	Minimalizácia váženého súčtu . . . . .	12
5.2.2	Implementácia <code>WeightedL1LInfModel</code> . . . . .	13
<b>6</b>	<b>Záver a diskusia</b>	<b>14</b>
<b>7</b>	<b>Prehľad kódu</b>	<b>15</b>

## 0 Úvod

V našej práci sa budeme venovať implementácii lineárnej regresie ako úlohy lineárneho programovania. Lineárna regresia je spôsob odhadovania závislej premennej  $y \in \mathbb{R}^n$  ako afinnej kombinácie nezávislých premenných  $x_1, \dots, x_n \in \mathbb{R}^n$ . Na meranie vzdialenosti medzi vektorom  $y$  a afinnej kombinácie budeme používať  $L^1$  a  $L^\infty$  normy, keďže práve pre tie sa dá tento problém naformulovať ako LP úloha.

V kapitole 1 sa venujeme matematickej formulácii LP úlohy a dokazovaniu jej optimality. V kapitole 2 vizualizujeme funkčnosť modelu na arbitrárnych 2D dátach `A04plotregres.npz`. Následne, v kapitole 3 sa venujeme predikovaniu ceny vína podľa dátového súboru `A04wine.csv`. Pre tieto predikcie následne spočítame koeficient determinácie v 4. Na záver, sekcia 5 popisuje našu implementáciu  $L^1$  a  $L^\infty$  regresii pre ľubovoľné dáta v programovacom jazyku Python. Tiež sa tam venujeme porovnávaniu správania týchto regresii a formulácii a implementácii minimalizovania váženej sumy týchto noriem.

# 1 Úloha A

Máme dané vektory  $y, x_1, x_2, \dots, x_k$ . Chceme nájsť parametre  $\beta_0, \beta_1, \dots, \beta_k$  také, aby pre vektor  $\hat{y} = \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k$ , boli normy  $\|y - \hat{y}\|_1$  a  $\|y - \hat{y}\|_\infty$  minimálne.

Vyjadríme vektor  $\hat{y}$  ako súčin matice a vektora  $\beta = (\beta_0, \beta_1, \dots, \beta_k)^T$ .

$$\hat{y} = \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k = \begin{pmatrix} | & | & | & \dots & | \\ \mathbf{1}_n & x_1 & x_2 & \dots & x_k \\ | & | & | & & | \end{pmatrix} \beta =: \mathbf{A}\beta$$

## 1.1 Minimalizovanie $L^1$ normy

Prevedieme problém zo zadania do tvaru:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \end{aligned}$$

Zaved'me si nový vektor premenných  $t \in \mathbb{R}^n$ , ktorým ohraničíme normu  $\|y - \mathbf{A}\beta\|_1$ .

$$-t \leq y - \mathbf{A}\beta \leq t$$

Pre obe ohraničenia, odseparujeme premenné od konštánt a prevedieme do maticového tvaru.

$$\begin{aligned} (\mathbf{A} \mid \mathbb{I}_n) \begin{pmatrix} \beta \\ t \end{pmatrix} &\geq y \\ (-\mathbf{A} \mid \mathbb{I}_n) \begin{pmatrix} \beta \\ t \end{pmatrix} &\geq -y \end{aligned}$$

Minimalizovanie  $L^1$  normy ako úloha lineárneho programovania vyzerá teda nasledovne.

$$\begin{aligned} \min \quad & (\mathbf{0}_{k+1}^T \mid \mathbf{1}_n^T) \begin{pmatrix} \beta \\ t \end{pmatrix} \\ \text{s.t.} \quad & \begin{pmatrix} \mathbf{A} \mid \mathbb{I}_n \\ -\mathbf{A} \mid \mathbb{I}_n \end{pmatrix} \begin{pmatrix} \beta \\ t \end{pmatrix} \geq \begin{pmatrix} y \\ -y \end{pmatrix} \\ & \beta \in \mathbb{R}^{k+1}, \quad t \geq \mathbf{0}_n \end{aligned} \tag{1}$$

### 1.1.1 Prípustnosť a optimalita

Dokážme, že (1) je úloha, ktorá nadobúda optimálne riešenie pre ľubovoľné vektory  $y, x_1, x_2, \dots, x_k$ . Nech  $|y| := (|y_1|, |y_2|, \dots, |y_n|)^T$  pre  $y = (y_1, y_2, \dots, y_n)^T$ . Ukážme prípustnosť zvolením  $\beta = \mathbf{0}_{k+1}$  a  $t = |y|$ :

$$\begin{aligned} \begin{pmatrix} \mathbf{A} \mid \mathbb{I}_n \\ -\mathbf{A} \mid \mathbb{I}_n \end{pmatrix} \begin{pmatrix} \mathbf{0}_{k+1} \\ |y| \end{pmatrix} &= \begin{pmatrix} |y| \\ |y| \end{pmatrix} \geq \begin{pmatrix} y \\ -y \end{pmatrix} \\ &|y| \geq \mathbf{0}_n \end{aligned}$$

Vidíme, že obe ohraňčenia platia, čiže  $(\mathbf{0}_{k+1}^T, |y|^T)^T$  je prípustné riešenie.

Optimalitu ukážeme zo silnej duality. Sformulujeme duálnu úlohu pre duálne premenné  $\alpha_1, \alpha_2 \in \mathbb{R}^n$ :

$$\begin{aligned} \max \quad & (y^T \mid -y^T) \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} \\ & (\mathbf{A}^T \mid -\mathbf{A}^T) \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} = \mathbf{0}_{k+1} \\ & (\mathbb{I}_n \mid \mathbb{I}_n) \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix} \leq \mathbf{1}_n \\ & \alpha_1, \alpha_2 \geq \mathbf{0}_n \end{aligned}$$

Vidíme, že táto úloha je prípustná pre  $\alpha_1 = \alpha_2 = \mathbf{0}_n$ . Z prípustnosti primárnej a duálnej úlohy teda vyplýva, že úloha (1) nadobúda optimálne riešenie pre ľubovoľnú voľbu počiatočných vektorov.

## 1.2 Minimalizovanie $L^\infty$ normy

Budeme používať rovnaké značenie pre predikovaný vektor hodnôt  $\hat{y} = \mathbf{A}\beta$  ako pri formulácii  $L^1$  normy. Zavedme si skalár  $\gamma \in \mathbb{R}$ , ktorým ohraňčíme normu  $\|y - \mathbf{A}\beta\|_\infty$ .

$$-\gamma \mathbf{1}_n \leq y - \mathbf{A}\beta \leq \gamma \mathbf{1}_n$$

Pre jednotlivé ohraňčenia odseparujeme premenné od konštánt a zapíšeme v maticovom tvare.

$$\begin{aligned} (\mathbf{A} \mid \mathbf{1}_n) \begin{pmatrix} \beta \\ \gamma \end{pmatrix} &\geq y \\ (-\mathbf{A} \mid \mathbf{1}_n) \begin{pmatrix} \beta \\ \gamma \end{pmatrix} &\geq -y \end{aligned}$$

Minimalizovanie  $L^\infty$  normy ako úloha lineárneho programovania vyzerá teda nasledovne.

$$\begin{aligned} \min \quad & (\mathbf{0}_{k+1}^T \mid 1) \begin{pmatrix} \beta \\ \gamma \end{pmatrix} \\ & \begin{pmatrix} \mathbf{A} \mid \mathbf{1}_n \\ -\mathbf{A} \mid \mathbf{1}_n \end{pmatrix} \begin{pmatrix} \beta \\ \gamma \end{pmatrix} \geq \begin{pmatrix} y \\ -y \end{pmatrix} \\ & \beta \in \mathbb{R}^{k+1}, \gamma \geq 0 \end{aligned} \tag{2}$$

### 1.2.1 Prípustnosť a optimalita

Podobný spôsobom ako vyššie ukážeme optimalitu (2). Nech  $\beta = \mathbf{0}_{k+1}$  a  $\gamma = |\tilde{y}|$ , kde  $|\tilde{y}| := |\max(y_1, y_2, \dots, y_n)|$  pre  $y = (y_1, y_2, \dots, y_n)^T$ :

$$\left( \begin{array}{c|c} \mathbf{A} & \mathbf{1}_n \\ \hline -\mathbf{A} & \mathbf{1}_n \end{array} \right) \left( \begin{array}{c} \mathbf{0}_{k+1} \\ |\tilde{y}| \end{array} \right) = \left( \begin{array}{c} |\tilde{y}| \mathbf{1}_n \\ |\tilde{y}| \mathbf{1}_n \end{array} \right) \geq \left( \begin{array}{c} y \\ -y \end{array} \right) \\ |\tilde{y}| \geq 0$$

Obe ohraničenia platia, čiže  $(\mathbf{0}_{k+1}^T, |\tilde{y}|)^T$  je prípustné riešenie. Sformulujme duálnu úlohu s duálnymi premennými  $\alpha_1, \alpha_2 \in \mathbb{R}^n$ :

$$\begin{aligned} \max \quad & (y^T \mid -y^T) \left( \begin{array}{c} \alpha_1 \\ \alpha_2 \end{array} \right) \\ & (\mathbf{A}^T \mid -\mathbf{A}^T) \left( \begin{array}{c} \alpha_1 \\ \alpha_2 \end{array} \right) = \mathbf{0}_{k+1} \\ & (\mathbf{1}_n^T \mid \mathbf{1}_n^T) \left( \begin{array}{c} \alpha_1 \\ \alpha_2 \end{array} \right) \leq 1 \\ & \alpha_1, \alpha_2 \geq \mathbf{0}_n \end{aligned}$$

Rovnako vidíme, že táto úloha je prípustná pre  $\alpha_1 = \alpha_2 = \mathbf{0}_n$ . Teda, zo silnej duality, úloha (2) nadobúda optimálne riešenie pre ľubovoľnú voľbu počiatočných vektorov.

## 2 Úloha B

### 2.1 Prevedenie úlohy LP do tvaru pre `scipy.optimize.linprog`

Metóda `linprog` z modulu `scipy.optimize` vyžaduje nasledujúci tvar úlohy LP:

$$\begin{aligned} \min \quad & c^T x \\ & A_{ub}x \leq b_{ub} \\ & A_{eq}x \leq b_{eq} \\ & l \leq x \leq u \end{aligned} \quad l, u \in (\mathbb{R} \cup \{\text{None}\})^n$$

Hodnota `None` vo vektoroch  $l, u$  značí neohraničenosť v danom smere. Upravme teda úlohy vyjadrené vyššie do predpísaného tvaru.

Pre  $L^1$  regresiu:

$$\begin{aligned} \min \quad & ( \mathbf{0}_{k+1}^T \mid \mathbf{1}_n^T ) \left( \frac{\beta}{t} \right) \\ & \left( \begin{array}{c|c} -\mathbf{A} & -\mathbb{I}_n \\ \hline \mathbf{A} & -\mathbb{I}_n \end{array} \right) \left( \frac{\beta}{t} \right) \leq \left( \frac{-y}{y} \right) \\ & \text{None} \leq \beta_i \leq \text{None} \quad i = 0, 1, \dots, k \\ & 0 \leq t_j \leq \text{None} \quad j = 1, \dots, n \end{aligned}$$

Pre  $L^\infty$  regresiu:

$$\begin{aligned} \min \quad & ( \mathbf{0}_{k+1}^T \mid 1 ) \left( \frac{\beta}{\gamma} \right) \\ & \left( \begin{array}{c|c} -\mathbf{A} & -\mathbf{1}_n \\ \hline \mathbf{A} & -\mathbf{1}_n \end{array} \right) \left( \frac{\beta}{\gamma} \right) \leq \left( \frac{-y}{y} \right) \\ & \text{None} \leq \beta_i \leq \text{None} \quad i = 0, 1, \dots, k \\ & 0 \leq \gamma \leq \text{None} \end{aligned}$$

Úlohy v zdrojovom kóde sú implementované práve v tomto tvare.

### 2.2 Implementovanie regresných LP úloh

Na implementáciu formulovaných LP úloh využívame tri knižnice:

- `numpy` - tvorenie matíc a vektorov, načítanie dát
- `scipy.optimize` - implementovaný LP solver
- `matplotlib.pyplot` na vykresľovanie grafov.

Dáta relevantné pre túto úlohu sú uložené v súbore `data/A04plotregres.npz`. Jedná sa o 16 bodov v  $\mathbb{R}^2$ , kde prvá súradnica reprezentuje nezávislú premennú (vektor týchto súradníc označíme  $x$ ) a druhá závislú premennú (označíme  $y$ ).

Vytvorme si potrebné štruktúry pre využitie metódy `scipy.optimize.linprog` pre LP formuláciu s  $L^1$  normou:

```
c = np.array([0,0] + [1] * len(x)) #objective function vector, two zeros
                                   #stand for betas
A = np.matrix([[1] * len(x), x]).transpose()
I = np.identity(len(x)) # Identity matrix

A_ub = np.block([[-A,-I], [A,-I]]) #creating a block matrix
b_ub = np.concatenate([-y, y]) #right side vector
bounds = [(None, None), (None, None)] + [(0, None) for _ in range(len(x))
                                           ] # bounds for variables
```

Pomocou solvera získame vektor optimálnych  $\beta$  koeficientov:

$$\beta_0^{(1)} \approx -9.8378, \beta_1^{(1)} \approx 2.1297$$

Podobne implementujeme  $L^\infty$  formuláciu:

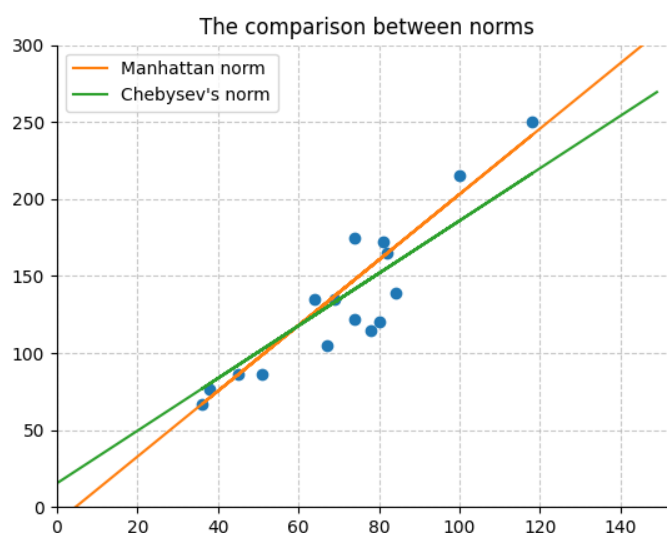
```
c_inf = np.array([0,0,1]) #objective function vector
A_inf = np.matrix([[1] * len(x), x]).transpose()
i_inf = np.array([[1] * len(x)]).transpose() # vector of ones

A_ub_inf = np.block([[-A_inf, -i_inf], [A_inf, -i_inf]]) # creating a
                                                         # block matrix
b_ub_inf = np.concatenate([-y, y]) #right side vector
bounds = [(None, None), (None, None), (0, None)]
```

Znovu, pomocou solvera získame vektor optimálnych  $\beta$  koeficientov:

$$\beta_0^{(\infty)} \approx 15.4545, \beta_1^{(\infty)} \approx 1.7045$$

Pomocou získaných  $\beta$  koeficientov vykreslíme regresné priamky spolu s pôvodnými dátami.





### 3 Úloha C

V tejto úlohe sa snažíme predikovať kvalitu vína, inšpirovaní prístupom Orleya Ashenfeltera k predikcii cien vína z Bordeaux.

Využívame dáta zo súboru `A04wine.csv` a aplikujeme modely  $L^1$  a  $L^\infty$  z úlohy A. Budeme využívať podobný postup ako v úlohe B. Na implementáciu formulovaných LP úloh využívame opäť:

- `pandas` - načítanie dát z `csv` súboru
- `numpy` - tvorenie matíc a vektorov
- `scipy.optimize` - implementovaný LP solver

Vyberieme z dát dané nezávislé premenné  $x$  a závislú premennú  $y$ :

```
y = data['Price']
x = data[['WinterRain', 'AGST', 'HarvestRain', 'Age', 'FrancePop']]
```

Z počtu nezávislých premenných získame rozmer vektora  $\beta$  (+1 kvôli konštantnému členu):

```
numberOfVariablesBeta = x.shape[1] + 1
```

Vytvoríme potrebné štruktúry pre zostavenie modelu normy  $L^1$ :

```
c = np.array([0]*numberOfVariablesBeta + [1] * len(x.values)) #
                                Objective function coefficients

ALeft = np.matrix([ [1] * len(x.values)]).transpose() # Coefficients for
                                beta0
ARigth = np.matrix(x.values) # Coefficients for other independent
                                variables beta
A = np.block([ALeft, ARigth]) # Concatenate coefficients of variables into
                                one matrix

I = np.identity(len(x.values)) # Identity matrix for intercept term
```

Naformulujeme problém a vyriešime pomocou `scipy.optimize.linprog`

```
A_ub = np.block([-A, -I], [A, -I])
b_ub = np.concatenate([-y, y])

solve = linprog(c, A_ub, b_ub, bounds = [(None, None)] *
                                numberOfVariablesBeta + [(0, None)])
```

Po vyriešení vyberieme z riešenia koeficienty:

```
betas = solve.x[:numberOfVariablesBeta]
```

Čo nám dá:

$$\begin{aligned}\beta_0^{(1)} &\approx -8.8801 \cdot 10^{-1}, \beta_1^{(1)} \approx 1.5793 \cdot 10^{-3}, \beta_2^{(1)} \approx 5.2130 \cdot 10^{-1} \\ \beta_3^{(1)} &\approx -4.5137 \cdot 10^{-3}, \beta_4^{(1)} \approx 1.1300 \cdot 10^{-2}, \beta_5^{(1)} \approx -2.2111 \cdot 10^{-5}\end{aligned}$$

Z týchto výsledkov môžeme usúdiť, že najviac pozitívne vplýva na cenu vína metrika *AGST* - *Average growing season temperature* a najsignifikantnejší negatívny vplyv má *dážď počas zberu*.

Ďalej zostrojíme relevantné štruktúry a naformulujeme LP pre  $L^\infty$  normu:

```
c_inf = np.array([0]*numberOfVariablesBeta + [1]) # Koeficienty cieľovej
                                                funkcie
A_inf = np.block([ALeft, ARigth]) # Koeficienty pre nezávislé premenné
                                   pre l_inf normu
i_inf = np.array([ [1] * len(x.values)]).transpose() # Koeficienty pre
                                                        intercept term pre l_inf normu

A_ub_inf = np.block([-A_inf, -i_inf], [A_inf, -i_inf])
b_ub_inf = np.concatenate([-y, y])
```

Vyriešime aj tento problém pomocou `scipy.optimize.linprog()` pre  $L^\infty$  normu a vyberieme  $\beta$  koeficienty:

```
solve_inf = linprog(c_inf, A_ub_inf, b_ub_inf, bounds=[(None, None)]*
                                                        numberOfVariablesBeta + [(0, None)])
betas_inf = solve_inf.x[:numberOfVariablesBeta]
```

Po čom dostaneme:

$$\beta_0^{(\infty)} \approx 3.4841, \beta_1^{(\infty)} \approx 8.3399 \cdot 10^{-4}, \beta_2^{(\infty)} \approx 6.0027 \cdot 10^{-1}$$

$$\beta_3^{(\infty)} \approx -3.3416 \cdot 10^{-3}, \beta_4^{(\infty)} \approx -2.3036 \cdot 10^{-2}, \beta_5^{(\infty)} \approx -1.1958 \cdot 10^{-4}$$

Vidíme, že aj regresia pomocou  $L^\infty$  normy odhaduje najväčší pozitívny vplyv metriky *AGST* a najväčší negatívny vplyv *dažďu počas zberu*. Zmenil sa však vplyv premennej *vek* (oproti prechádzajúcemu modelu) z pozitívneho na negatívny.

## 4 Úloha D

Vytvorme funkciu `r_squared(x, y, beta)` - kde `x` je matica vektorov nezávislých premenných, `y` je vektor závislej premennej, `beta` je vektor optimálnych  $\beta$  koeficientov získaných regresiou - ktorá bude počítať  $R^2$  koeficient podľa definície:

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad \hat{y} = \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k, \quad \bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$$

```
def r_squared(x: np.ndarray, y: np.ndarray, beta: np.ndarray) -> float:
    # calculate y-hat and mean of y vector
    y_hat = beta[0] + np.dot(x, beta[1:])
    y_mean = np.mean(y)

    res1 = 0      # partial result for the numerator in the formula
    res2 = 0      # partial result for the denominator in the formula

    # calculate the sums
    for i in range(len(y)):
        res1 += (y[i] - y_hat[i]) ** 2
        res2 += (y[i] - y_mean) ** 2

    # calculate the R^2 coefficient
    result = 1 - (res1 / res2)
    return result
```

Implementujeme metódu na dátach `A04wine.csv`. Načítame dáta pomocou `pandas.read_csv()`, rozdelíme ich do premenných (rovnako ako v predošlých úlohách):

```
data = pd.read_csv('data/A04wine.csv')
y = data['Price']
x = data[['WinterRain', 'AGST', 'HarvestRain', 'Age', 'FrancePop']]
```

Podobne ako vyššie, vyriešime potrebné LP problémy pre načítané dáta a vypočítame  $R^2$  koeficient:

```
betas = solve.x[:numberOfVariablesBeta]
betas_inf = solve_inf.x[:numberOfVariablesBeta]

r_squared(x, y, betas)
r_squared(x, y, betas_inf)
```

Vypočítané príslušné R-kvadráty teda sú:

$$R_{(1)}^2 \approx 0.78813$$

$$R_{(\infty)}^2 \approx 0.80649$$

Z toho môžeme usúdiť, že náš model sa dá považovať za relatívne vhodný pre tieto dáta. Tiež vidíme, že regresia pomocou Chebyshevovej normy lepšie zachytáva rozptyl dát.

## 5 Úloha E

### 5.1 Spracovanie všeobecnej triedy pre $L^1$ a $L^\infty$ regresiu

Vypracovali sme modul `Model` pre počítanie  $L^1$  a  $L^\infty$  regresie z ľubovoľných číselných dát, ktorý využíva LP formulácie popísané vyššie. Konkrétne `L1Model` využíva formuláciu na minimalizovanie  $L^1$  normy a `LInfModel` minimalizuje  $L^\infty$  normu. Príklad použitia tohto modelu sa nachádza v `model_demonstration.ipynb`. Následne opíšeme jednotlivé metódy jednotlivých modelov.

`Model.__init__(dependent_vect, independent_vect)`

Konštruktor triedy, spoločný pre oba modely, vytvorí inštanciu, ktorá si drží dáta a vie na nich vykonávať operácie popísané nižšie.

Argumenty:

- `dependent_vect: np.ndarray` - vektor závislých premenných
- `independent_vect: np.ndarray` - matica, ktorej riadky sú vektory nezávislých premenných

`Model.solve()`

Metóda, ktorá vyrieši regresnú LP úlohu na daných dátach. `L1Model.solve()` rieši minimalizáciou  $L^1$  normy a `LInfModel.solve()`, rieši minimalizáciou  $L^\infty$  normy.

Vracia:

- `np.ndarray` - vektor optimálnych  $\beta$  premenných

Po zavolaní tejto metódy si inštancia uloží vektor optimálnych  $\beta$  premenných do atribútu `self._beta`, potrebné pre metódy popísané nižšie.

`Model.r2()`

Vypočíta  $R^2$  koeficient pre dané dáta a vypočítaný vektor  $\beta$ .

Vracia:

- `float` - výsledný  $R^2$  koeficient

`Model.visualize()`

Ak je počet nezávislých premenných 1 alebo 2, táto metóda vykreslí graf dát spolu s vypočítanou regresnou priamkou, resp. rovinou.

Vracia:

- `bool` - úspešnosť vizualizácie, kde `False` označuje, že nezávislých premenných je viac ako 2, čiže nie je možné vykresliť graf

## 5.2 Porovnanie použitia $L^1$ a $L^\infty$ lineárnej regresie

Nasledujúce tvrdenia popisujú len naše pozorovania správania sa jednotlivých lineárnych regresii na generovaných dátach

Vyššie v sekcii 1 sme ukázali, že implementácie lineárnej regresie pomocou merania vzdialenosti  $L^1$  a  $L^\infty$  normou majú optimálne riešenie, pre ľubovoľné vstupné dáta. Snažili sme sa odpozorovať, ako sa jednotlivé prístupy odlišujú pre nejaké konkrétne dáta.

V dátach, v ktorých je výrazná lineárna závislosť, minimalizovanie  $L^1$  normy veľmi dobre zachytáva práve tento lineárny vzťah, aj v prítomnosti odľahlých dát - outlierov. Toto správanie vie ale viesť aj k tzv. *overfittingu*. Model príliš tesne zachytáva takéto správanie, čo môže viesť k horším odhadom pre budúce pozorovania.

Na druhej strane minimalizovanie  $L^\infty$  normy je veľmi ovplyvňované outliermi. Aj pre „jasné“ lineárne dáta s nejakými chybnými pozorovaniami, tieto dátové body výrazne odklonia regresnú priamku/nadrovinu.

### 5.2.1 Minimalizácia váženého súčtu

Toto správanie  $L^\infty$  lineárnej regresie môžeme využiť na zníženie *overfittingu*  $L^1$  lineárnej regresie. Jeden z možných prístupov môže byť napríklad pomocou minimalizácie váženej sumy  $\omega \|y - \hat{y}\|_1 + (1 - \omega) \|y - \hat{y}\|_\infty$ ,  $\omega \in [0; 1]$ . Formulovaná LP úloha vyzerá nasledovne (značenie sme prebrali z (1) a (2)):

$$\begin{aligned} \min \quad & \left( \begin{array}{c|c|c} \mathbf{0}_{k+1}^T & \omega \mathbf{1}_n^T & (1 - \omega) \end{array} \right) \left( \begin{array}{c} \beta \\ t \\ \gamma \end{array} \right) \\ & \left( \begin{array}{c|c|c} \mathbf{A} & \mathbb{I}_n & \mathbf{0}_n \\ -\mathbf{A} & \mathbb{I}_n & \mathbf{0}_n \\ \hline \mathbf{A} & \mathbf{0}_{n \times n} & \mathbf{1}_n \\ -\mathbf{A} & \mathbf{0}_{n \times n} & \mathbf{1}_n \end{array} \right) \left( \begin{array}{c} \beta \\ t \\ \gamma \end{array} \right) \geq \left( \begin{array}{c} y \\ -y \\ y \\ -y \end{array} \right) \\ & \beta \in \mathbb{R}^{k+1}, \quad t \geq \mathbf{0}_n, \quad \gamma \geq 0 \end{aligned}$$

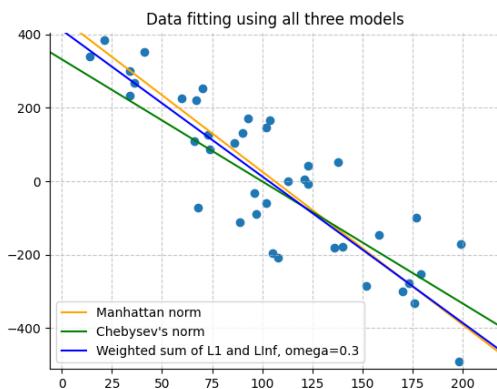
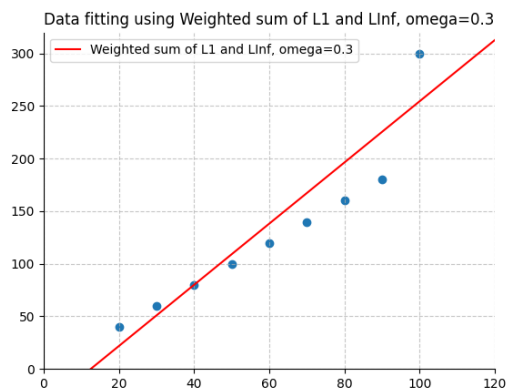
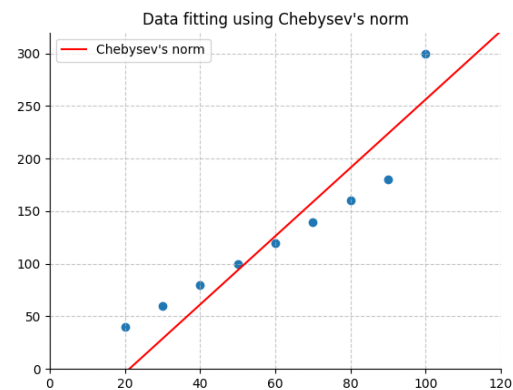
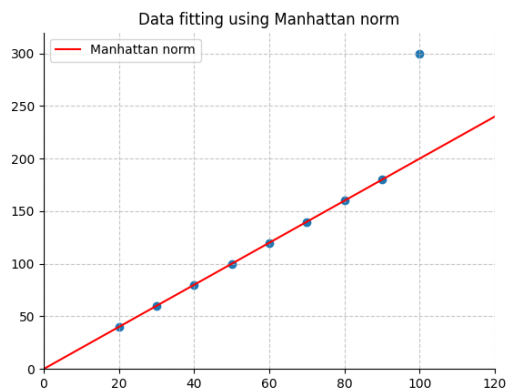
Podobným spôsobom ukážeme, že táto úloha nadobúda optimálne riešenie. Sformulujme duálnu úlohu:

$$\begin{aligned} \text{Nech } \alpha &= \left( \begin{array}{c} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{array} \right), \quad \alpha_{1,2,3,4} \in \mathbb{R}^n \\ \max \quad & \left( \begin{array}{c|c|c|c} y^T & -y^T & y^T & -y^T \end{array} \right) \alpha \\ & \left( \begin{array}{c|c|c|c} \mathbf{A}^T & -\mathbf{A}^T & \mathbf{A}^T & -\mathbf{A}^T \end{array} \right) \alpha = \mathbf{0}_{k+1} \\ & \left( \begin{array}{c|c|c|c} \mathbb{I}_n & \mathbb{I}_n & \mathbf{0}_{n \times n} & \mathbf{0}_{n \times n} \end{array} \right) \alpha \leq \omega \mathbf{1}_n \\ & \left( \begin{array}{c|c|c|c} \mathbf{0}_n^T & \mathbf{0}_n^T & \mathbf{1}_n^T & \mathbf{1}_n^T \end{array} \right) \alpha \leq 1 - \omega \\ & \alpha \geq \mathbf{0}_{4n} \end{aligned}$$

Vidíme, že primárna úloha je prípustná pre  $\beta = \mathbf{0}_{k+1}$ ,  $t = |y|$ ,  $\gamma = |\hat{y}|$  (využitím značenia ako v 1.1.1 a 1.2.1) a duálna úloha je prípustná pre  $\alpha = \mathbf{0}_{4n}$ , teda zo silnej duality obe riešenia nadobúdajú optimálne riešenie.

## 5.2.2 Implementácia `WeightedL1LInfModel`

Takáto lineárna regresia je implementovaná v triede `WeightedL1LInfModel`. Jej používanie je rovnaké ako pri predchádzajúcich implementáciách. Jediná zmena je pre metódu `WeightedL1LInfModel.solve(omega)`, ktorá teraz očakáva parameter `omega` : `float` v intervale  $[0; 1]$ .



Porovnanie správania sa jednotlivých regresii, prvé tri grafy zobrazujú rovnaké lineárne dáta s jedným outlierom

## 6 Záver a diskusia

V našom projekte sme sa venovali matematickej formulácii a implementácii lineárnej regresie minimalizovaním  $L^1$  a  $L^\infty$  noriem. Vizualizovali sme funkčnosť implementácie na dátach `A04plotregres.npz` a pre dáta `A04wine.csv` sme regresiou predikovali budúcu cenu vína a zisťovali sme, ktoré parametre na ňu najviac vplyvajú. Takisto sme pre túto predikciu spočítali  $R^2$  koeficient, ktorý ukázal relatívnu vhodnosť nášho modelu. Nakoniec sme predstavili implementáciu týchto regresných modelov v jazyku Python pre ľubovoľné číselné dáta a mierne sme analyzovali správanie sa jednotlivých modelov. Nakoniec sme aj sformulovali a implementovali model pre minimalizovanie váženej sumy noriem.

Myslíme si, že naše modely sú jednoduchým nástrojom pre počítanie lineárnej regresie. Ich výhody oproti klasickej  $L^2$  lineárnej regresii spočívajú aj v ich robustnosti, čiže odľahlé dáta v nich majú menšiu váhu. Ako ďalšie pokračovanie projektu by sme mohli skúmať charakteristiky jednotlivých modelov a zistiť, pre aké dáta je lepšie použiť jednotlivé normy. Tiež by sme sa mohli zaoberať ich časovou komplexitou (napríklad aj v porovnaní s  $L^2$  lineárnou regresiou) a všeobecnou interpretáciou výsledných  $\beta$  koeficientov pre oba prístupy.

## 7 Prehľad kódu

```
/
├── code
│   ├── data
│   │   ├── A04plotregres.npz - arbitrárne 2D dáta využívané v 2
│   │   └── A04wines.csv - dátový súbor štatistík vín využívaný v 3 a 4
│   ├── models
│   │   └── models.py - implementované triedy pre  $L^1$  a  $L^\infty$  lineárnu regresiu
│   ├── task_b.py - kód k sekcii 2
│   ├── task_c.py - kód k sekcii 3
│   ├── task_d.py - kód k sekcii 4
└── report.pdf - report celého projektu
```