Trabalho de Introdução à Ciência de Dados

Equipe:

Fernanda Luísa Silva Gomes
João Lucas Duim

Questão 1

O método escolhido é "Modelos de Regressão".

Questão 2

O termo "regressão" foi utilizado pela primeira vez por Francis Galton em um estudo que demonstrou que a altura dos filhos não tende a refletir a altura dos pais, mas tende a regredir para a média da população. A regressão é o processo matemático no qual os coeficientes associados às variáveis (atributos) são calculados de modo a minimizar o erro do modelo com relação aos dados de treinamento. Uma vez que o modelo esteja calculado, é utilizado para predizer valores a partir de informações dos atributos dos dados, ou seja, dado um conjunto de atributos, o modelo gera uma aproximação da variável dependente. Alguns dos principais métodos de regressão são linear, polinomial, Ridge e Lasso.

Regressão Linear:

A regressão linear é um processo supervisionado em que um modelo linear é adotado para modelar os dados. Modelos lineares possuem coeficientes associados a cada atributo: coeficientes que possuem maior magnitude indicam que o atributo correspondente tem maior importância no processo de predição, enquanto coeficientes com magnitude próxima de zero indicam atributos pouco relevantes. Em um processo de regressão linear, atributos devem ser normalizados e outliers devem ser evitados, bem como o uso de atributos altamente correlacionados. A regressão é feita (cálculo dos coeficientes) utilizando-se os dados de treinamento, e o modelo gerado é avaliado nos dados de teste. Como o modelo muda dependendo do dado de treinamento, o que se faz é utilizar um mecanismo chamado validação cruzada: um processo que consiste em dividir os dados originais (conhecidos) em conjuntos de dados de treinamento e de teste. Os conjuntos são gerados selecionando amostras randômicas para cada um desses tipos. Cada conjunto de treinamento gerado dá origem a um modelo. A previsão é feita a partir da média das previsões realizadas por cada modelo. A eficácia (erro) também é calculada como a média da eficácia desses modelos.

Definido o tipo do modelo e o dado de treinamento, os coeficientes são calculados de modo a minimizar o erro entre o modelo e os dados. Existem diversas alternativas matemáticas para calcular os coeficientes: métodos de otimização e método dos mínimos quadrados. Modelos de regressão linear são frequentemente ajustados usando a abordagem dos mínimos quadrados, mas que também pode ser montada de outras maneiras, tal como minimizando a "falta de ajuste" em alguma outra norma (com menos desvios absolutos de regressão), ou através da minimização da função de perda de uma versão penalizada dos mínimos quadrados, como as regressões de Ridge e de Lasso. Por outro lado, a abordagem de mínimos quadrados pode ser

utilizada para ajustar a modelos que não são lineares. Assim, embora os termos "mínimos quadrados" e "modelo linear" estejam intimamente ligados, eles não são sinônimos.

Regressão Polinomial:

Os modelos de regressão polinomial são geralmente ajustados usando o método dos mínimos quadrados. O método dos mínimos quadrados minimiza a variância dos estimadores imparciais dos coeficientes, nas condições do teorema de Gauss-Markov. O método dos mínimos quadrados foi publicado em 1805 por Legendre e em 1809 por Gauss. O primeiro projeto de um experimento para regressão polinomial apareceu em um artigo de Gergonne em 1815. No século XX, a regressão polinomial desempenhou um papel importante no desenvolvimento da análise de regressão, com maior ênfase em questões de design e inferência. Mais recentemente, o uso de modelos polinomiais foi complementado por outros métodos, com modelos não polinomiais apresentando vantagens para algumas classes de problemas.

Regressão polinomial é uma forma de análise de regressão em que a relação entre a variável independente x e a variável dependente y é modelado como um polinômio de grau n em x. A regressão polinomial se ajusta a uma relação não linear entre o valor de x e a média condicional correspondente de y, denotada por $E(y|x)$. Embora a regressão polinomial se ajuste a um modelo não linear aos dados, como um problema de estimação estatística, é linear, no sentido de que a função de regressão $E(y|x)$ é linear nos parâmetros desconhecidos que são estimados a partir dos dados . Por esse motivo, a regressão polinomial é considerada um caso especial de regressão linear múltipla.

Embora a regressão polinomial seja tecnicamente um caso especial de regressão linear múltipla, a interpretação de um modelo de regressão polinomial ajustado requer uma perspectiva um pouco diferente. Freqüentemente, é difícil interpretar os coeficientes individuais em um ajuste de regressão polinomial, uma vez que os monômios subjacentes podem ser altamente correlacionados. Por exemplo, x e x^2 têm correlação em torno de $0,97$ quando x é uniformemente distribuído no intervalo $(0, 1)$. Embora a correlação possa ser reduzida usando polinômios ortogonais , geralmente é mais informativo considerar a função de regressão ajustada como um todo. Faixas de confiança pontuais ou simultâneas podem então ser usadas para fornecer uma noção da incerteza na estimativa da função de regressão.

Regressão de Ridge:

A regressão Ridge é um método de estimar os coeficientes de modelos de regressão múltipla em cenários onde as variáveis independentes são altamente correlacionadas. Tem aplicações em campos como econometria, química e engenharia.

A teoria foi introduzida pela primeira vez por Hoerl e Kennard em 1970 em seus artigos Technometrics "Regressões RIDGE: estimativa enviesada de problemas não ortogonais” e “Regressões RIDGE: aplicações em problemas não ortogonais”. Este foi o resultado de dez anos de pesquisa no campo da análise de cristas.

A regressão Ridge foi desenvolvida como uma possível solução para a imprecisão dos estimadores de mínimos quadrados quando os modelos de regressão linear têm algumas variáveis independentes multicolineares (altamente correlacionadas) - criando um estimador de regressão ridge (RR). Isso fornece uma estimativa mais precisa dos parâmetros da crista, uma vez que sua variância e o estimador de erro quadrático médio são geralmente menores do que os estimadores de mínimos quadrados derivados anteriormente.

Regressão de Lasso:

O Lasso (least absolute shrinkage and selection operator) é um método de análise de regressão que realiza seleção e regularização de variáveis para aprimorar a precisão da previsão e a interpretabilidade do modelo estatístico resultante. Ele seleciona um conjunto reduzido de covariáveis conhecidas para uso em um modelo. Ele atinge seus objetivos ao forçar a soma do valor absoluto dos coeficientes de regressão a ser menor que um valor fixo, o que força certos coeficientes a zero, excluindo-os da previsão de impacto. Essa ideia é semelhante à regressão de Ridge, que também reduz o tamanho dos coeficientes; no entanto, a regressão de Ridge tende a definir muito menos coeficientes para zero.

Questão 3

Introdução:

Um pêndulo simples é um sistema composto por um fio inextensível e de massa desprezível, preso a um suporte, cuja extremidade contém uma massa pontual, que pode movimentar-se livremente. O objetivo é utilizar computacionalmente o método de regressão linear por mínimos quadrados a fim de ajustar os dados da melhor forma possível em uma reta, de modo a constatar experimentalmente a teoria física que rege o movimento de um pêndulo simples. Adicionalmente, ser capaz de estimar com boa aproximação a aceleração da gravidade no local do experimento.

As bases de dados coletadas aqui (simple_pendulum_data_train.csv) e aqui (simple_pendulum_data_test.csv) consistem de noventa linhas e duas colunas, uma contendo o comprimento do fio no experimento e a outra contendo o tempo de uma oscilação completa da massa (período). Note que ambas as bases de dados registram o experimento para os mesmos comprimentos do fio do pêndulo, com diferentes precisões de medida do período. Além disso, os experimentos devem ter sido feitos em locais diferentes, sob diferentes acelerações da gravidade.

O método de mínimos quadrados busca encontrar a melhor reta que se ajusta aos dados, de modo a minimizar a soma dos quadrados das diferenças entre o valor estimado e os dados observados. Um requesito para a aplicação desse método é que o erro seja distribuído aleatoriamente. Além disso, o modelo deve ser linear nos parâmetros, ou seja, as variáveis devem apresentar uma relação linear entre si.

Sendo assim, seja n pontos (t_i, b_i) e deseja-se encontrar C e D tal que $C + DT_i = b_i$. Temos:

$$A = \begin{bmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_n \end{bmatrix}$$

$$\hat{x} = (A^T A)^{-1} A^T b$$

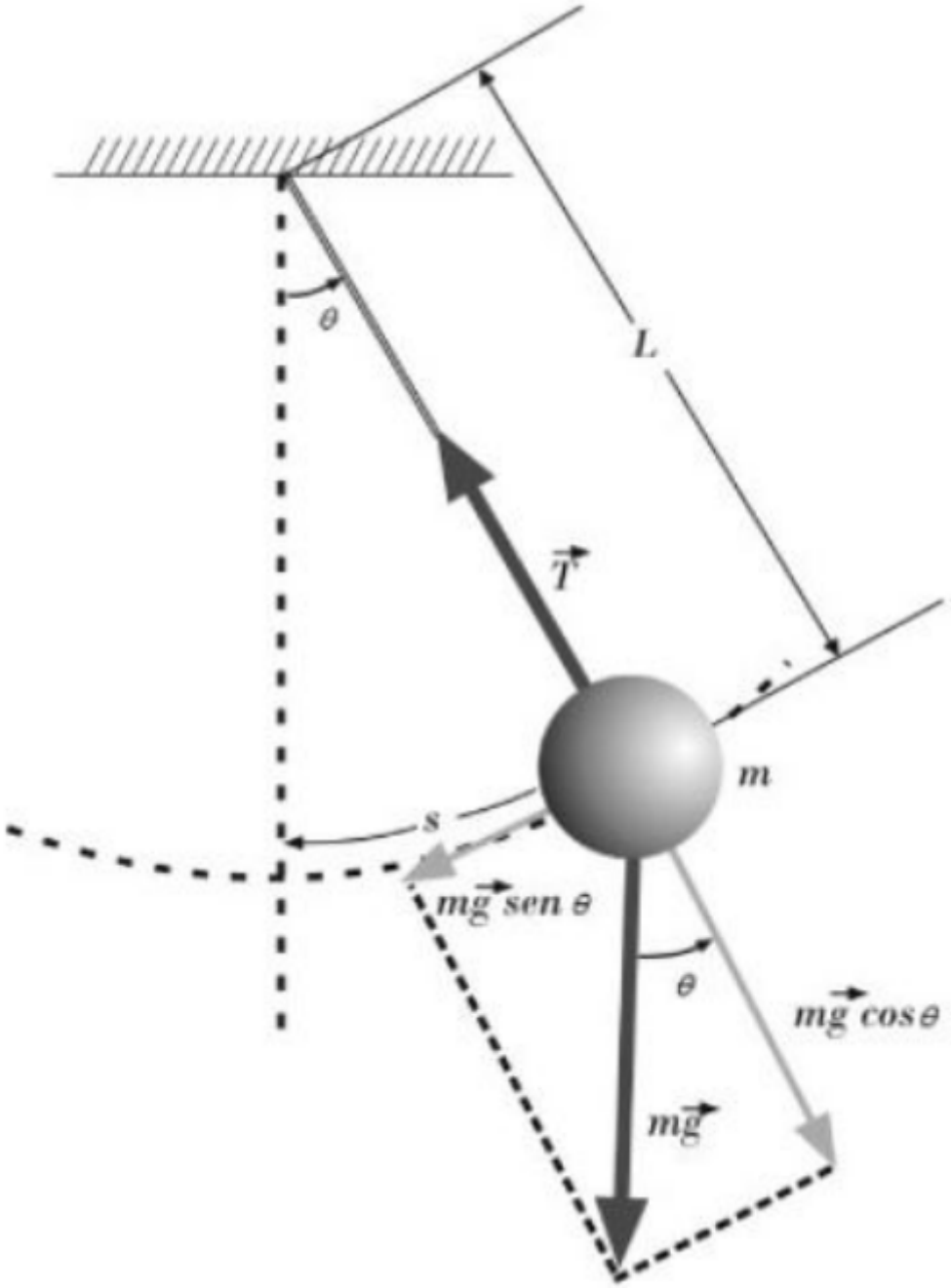
Além disso:

$$A^T A = \begin{bmatrix} n & \sum t_i \\ \sum t_i & \sum t_i^2 \end{bmatrix}$$

$$A^T b = \begin{bmatrix} \sum b_i \\ \sum t_i b_i \end{bmatrix}$$

Embasamento teórico:

Deslocando a massa de um pequeno ângulo, pode-se decompor as forças que atuam na partícula (tração e peso) nas direções normal e tangencial à trajetória. Adotemos o sentido anti-horário como positivo no deslocamento sobre a trajetória curva.



Nos cálculos a seguir, substituiremos s da figura por x e L da figura por l. Temos, então, as seguintes variáveis:

x é o comprimento da trajetória da partícula sob o ângulo θ
l é o raio da trajetória da partícula, ou seja, o comprimento do fio;
m é a massa da partícula;
θ é o ângulo máximo de deslocamento da partícula em relação à vertical (posição de equilíbrio);
g é o módulo da aceleração da gravidade local;
T é o módulo da tração exercida pelo fio sobre a partícula.

Como há equilíbrio da partícula na direção normal à trajetória, a componente do peso nessa direção se iguala em módulo à tração exercida no fio:

$$T = mg\sin\theta$$

Assim, a aceleração que a partícula sofre é proveniente da componente tangencial do peso (força resultante). Pela 2° Lei de Newton, temos:

$$m\frac{d^2x}{dt^2} = -mg\sin\theta$$

$$\frac{d^2x}{dt^2} = -g\sin\theta$$

$$\frac{d^2x}{dt^2} + g\sin\theta = 0$$

Como θ é um ângulo pequeno, $|\theta| << 1 \implies \sin\theta \approx \theta$. Além disso, sabemos que:

$$x = l \cdot \theta$$

$$\frac{d^2x}{dt^2} = l\frac{d^2\theta}{dt^2}$$

Temos, então, a seguinte equação:

$$\frac{d^2\theta}{dt^2} + \frac{g}{l}\theta = 0$$

Note a semelhança com a equação do oscilador harmônico:

$$\frac{d^2x}{dt^2} + \omega^2 x = 0$$

Podemos, então, dizer que o pêndulo simples realiza um movimento muito próximo de um MHS (Movimento Harmônico Simples) para um ângulo pequeno θ, com pulsação dada por:

$$\omega = \sqrt{\frac{g}{l}}$$

Além disso, sendo T o período de um MHS, sabemos que a pulsação desse MHS é dada por:

$$\omega = \frac{2\pi}{T}$$

Logo:

$$T = 2\pi\sqrt{\frac{l}{g}}$$

Essa relação fornece o período em função do comprimento do fio do pêndulo.

$$T^2 = \frac{4\pi^2}{g} \cdot l$$

Finalmente, nota-se a dependência linear de T^2 em relação a l, que será explorada a seguir.

Vale ressaltar que, embora não seja previsto teoricamente (e idealmente) um coeficiente independente de l, ajustaremos a seguir um modelo com esse coeficiente. Qualquer dado experimental está sujeito a erros de medição e, na prática, a partícula sofre atrito com o ar, não é puntiforme, o fio possui uma massa, entre outros fatores que distanciam a realidade da idealidade. Tal coeficiente fornece informações bastante úteis a respeito dos erros envolvidos no procedimento, e por isso é importante incluí-lo. Isso justifica a utilização do referido modelo, ainda que aqui não seja feita uma análise para além da idealidade.

Código Python Utilizado:

Realizaremos os seguintes procedimentos no código apresentado:

(1) Leia a base de dados de treino.
(2) Separe os dados em dois arrays, t e b.
(3) ones <- vetor de 1 do mesmo tamanho de t.
(4) A <- junção de ones e t em um único array.
(5) A <- A transposta.
(6) AtA <- A transposta vezes A.
(7) Atb <- A transposta vezes b.
(8) x_hat <- solução do sistema linear (AtA)x = Atb.
(9) t_linspace <- sequencia de números uniformemente espalhados no intervalo [min(t), max(t)].
(10) least_square <- primeiro valor de x_hat + segundo valor de x_hat *t.*
(11) *least_square_plot <- primeiro valor de x_hat + segundo valor de x_hat* t_linspace.
(12) Plote a base de dados de teste e os dados t_linspace e least_square_plot.
(13) Avalie a performance do modelo.

Inicialmente, importaram-se os pacotes necessários para a resolução do problema escolhido (numpy, pandas, matplotlib e sklearn). Posteriormente, configurou-se e personalizou-se o ambiente de plotagem.

In [1]:
```
#import das bibliotecas a serem utilizadas
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn import metrics

#na variável params, definiu-se os parâmetros desejados para os plots a serem
params = {'legend.fontsize': 20,
          'figure.figsize': (20, 15),
          'axes.labelsize': 20,
          'axes.titlesize':20,
          'xtick.labelsize':20,
          'ytick.labelsize':20}

#a função style.use da biblioteca matplotlib.pyplot define o estilo desejado
plt.style.use('ggplot')

#a função rcParams.update da biblioteca matplotlib.pyplot atualiza os parâmet
plt.rcParams.update(params)
```

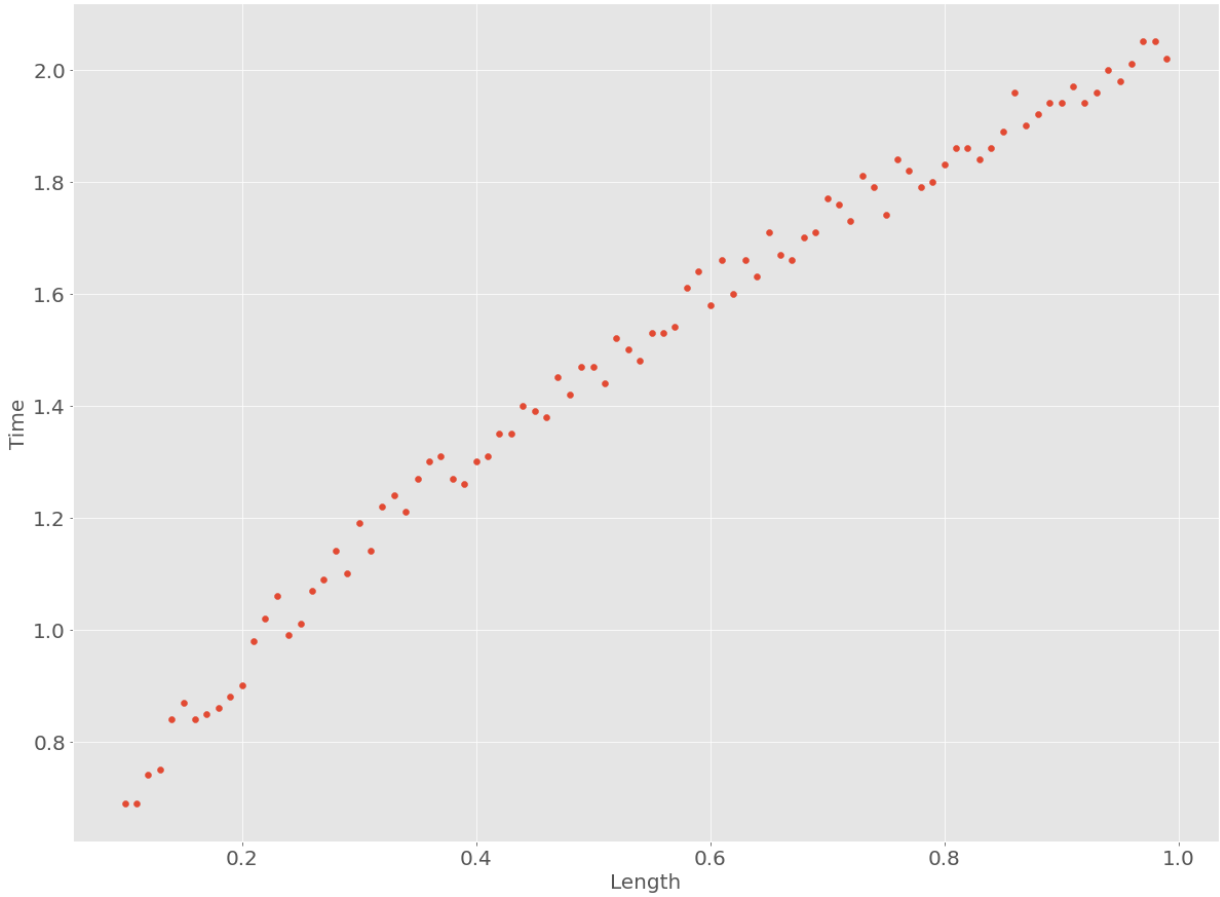
Os dados de treino foram lidos e analisados. Adicionalmente, os dados referentes ao período e ao comprimento foram devidamente extraídos da base de treino utilizada.

In [2]:
```
#por meio da função read_csv da biblioteca pandas, realizou-se a leitura da b
data = pd.read_csv('simple_pendulum_data_train.csv').dropna()

#print da descrição básica da base de dados utilizando o método describe da b
print(data.describe())

#armazenou-se em periodo os valores da coluna 'time' do dataframe utilizando
periodo = data['time'].values

#armazenou-se em comprimento os valores da coluna 'length' do dataframe utili
comprimento = data['length'].values
```

```
          length       time
count  90.000000  90.000000
mean    0.545000   1.478556
std     0.261247   0.382612
min     0.100000   0.690000
25%     0.322500   1.212500
50%     0.545000   1.525000
75%     0.767500   1.807500
max     0.990000   2.050000
```

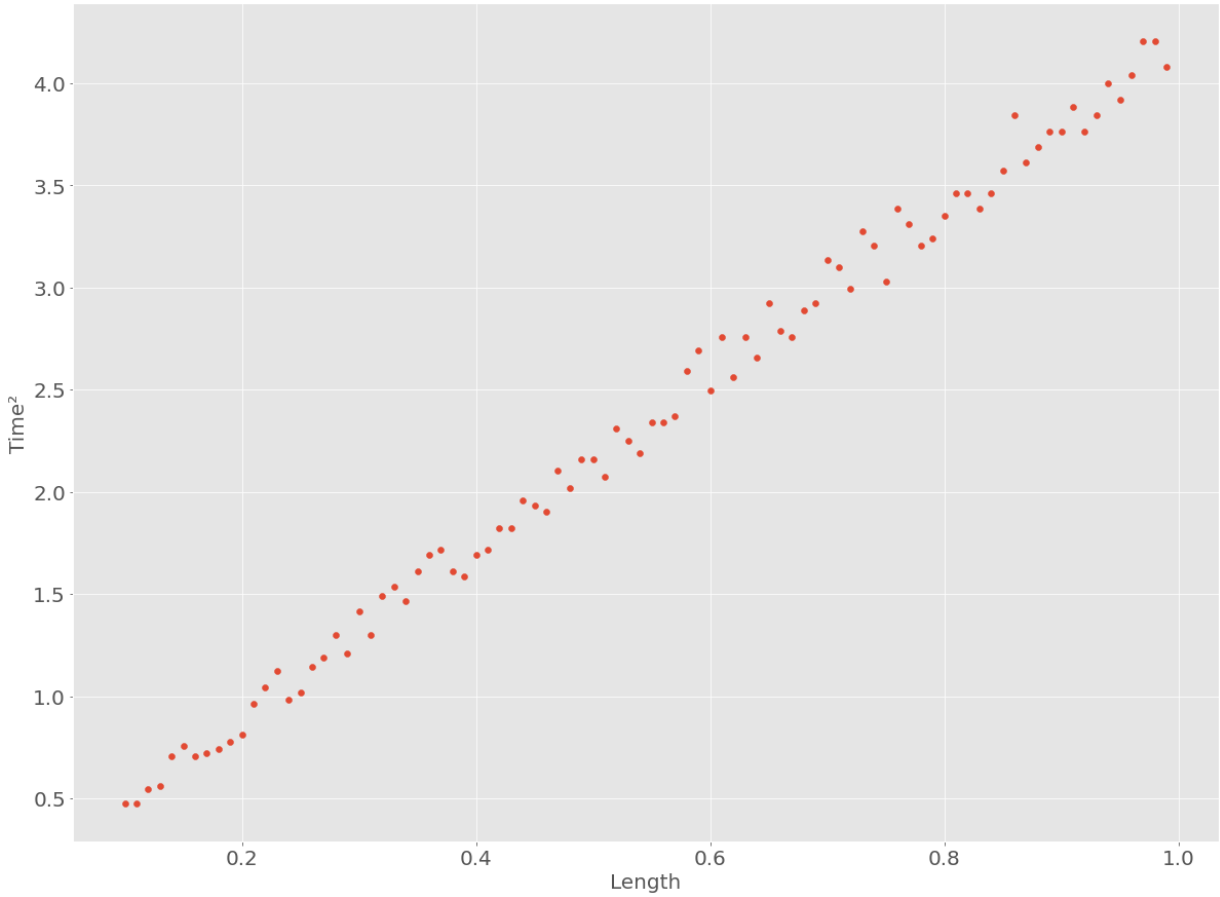
De modo a aumentar a compreensão sobre os dados e a relação entre eles, plotou-se o gráfico Time vs Length.

In [3]:
```
#criação de uma figura em que no eixo X temos o comprimento e no eixo Y o per
plt.figure()
plt.scatter(comprimento, periodo)
plt.xlabel('Length')
plt.ylabel('Time')
plt.show()
```



Como esperado, o comprimento e o período não possuem uma relação linear entre si. Isso ocorre com o comprimento e o quadrado do periodo, conforme visto anteriormente.

In [15]:
```
#armazenou-se em t os valores do comprimento
t = comprimento
#armazenou-se em b os valores do período ao quadrado
b = (periodo) ** 2

#criação de uma figura em que no eixo X temos o comprimento e no eixo Y o per
plt.figure()
plt.scatter(t, b)
plt.xlabel('Length')
plt.ylabel('Time²')
plt.show()
```



A seguir, ajusta-se o modelo aos dados de treino e obtêm-se a equação da reta ajustada e a sua plotagem juntamente aos dados.

In [5]:
```
#cria um array de 1 com o tamnho de t utilizando a função ones da biblioteca
ones = np.ones(len(t))

#une o array de 1 e t, depois transpõe o array unificado utilizando a função
A = np.array([ones, t]).T
```

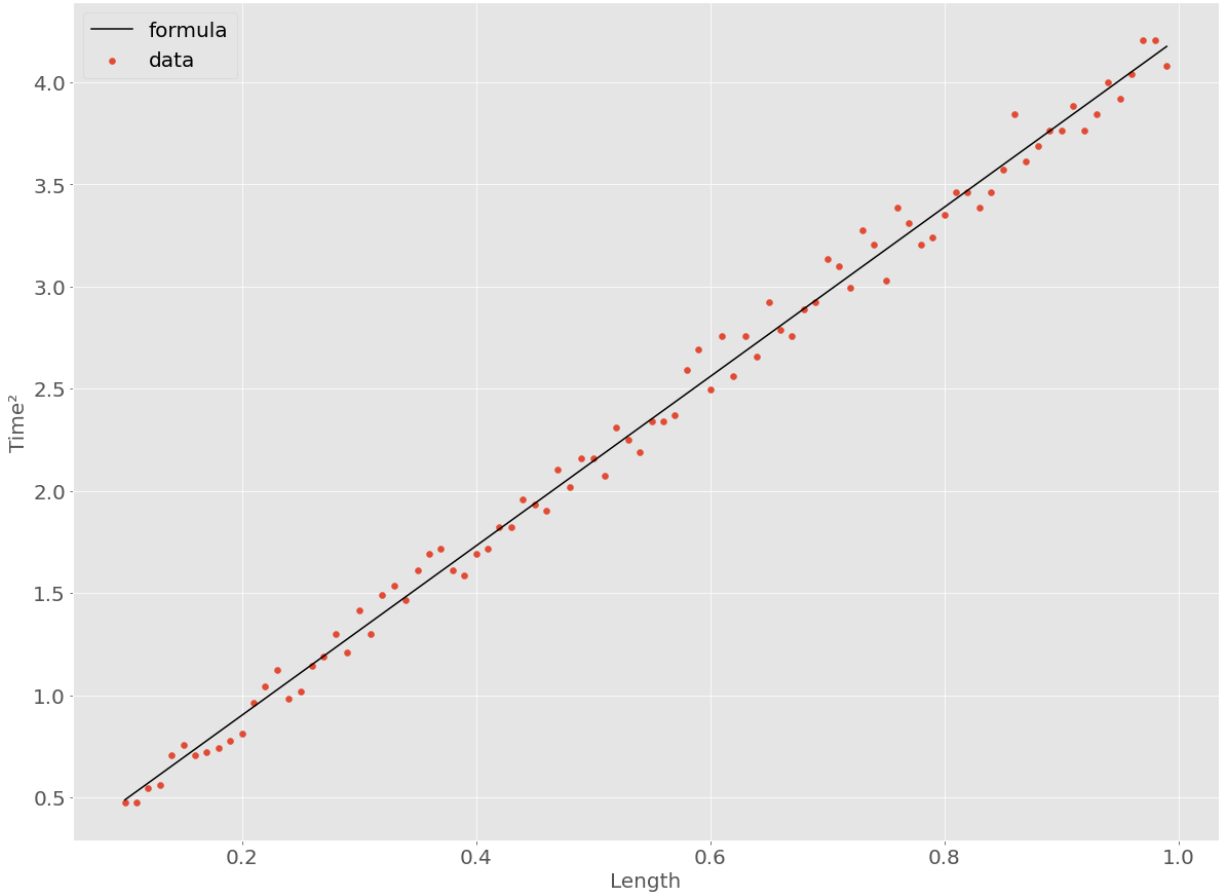
In [6]:
```
#multiplica A transposta por A utilizando o operador @ da biblioteca numpy
AtA = A.T @ A
#multiplica A transposta por b utilizando o operador @ da biblioteca numpy
Atb = A.T @ b

#resolve o sistema (AtA)x = (At)b utilizando a função linalg.solve da bibliot
x_hat = np.linalg.solve(AtA, Atb)

#print da equação da reta ajustada
print(f'Equação da reta ajustada: T^2 = {round(x_hat[1],2)} * l + {round(x_ha
```

```
Equação da reta ajustada: T^2 = 4.14 * l + 0.07
```

In [7]:
```
#constroi sequência de números uniformemente espaçados no intervalo [min(t),
#biblioteca numpy
t_linspace = np.linspace(np.min(t), np.max(t), 101)

#predições
least_square = x_hat[0] + x_hat[1] * t
least_square_plot = x_hat[0] + x_hat[1] * t_linspace
```

In [8]:
```
#criação de uma figura com os dados de treino e a reta que minimiza o erro qu
```

```
#e no eixo Y o periodo ao quadrado
fig, ax = plt.subplots()
plt.scatter(t, b, label='data')
plt.plot(t_linspace, least_square_plot, color='k', label='formula')
plt.xlabel('Length')
plt.ylabel('Time²')
plt.legend()
plt.show()
```



Verifica-se o grau de performance do modelo sobre os próprios dados de treino.

In [9]:
```
#calculo do erro quadrático médio sobre os próprios dados de treino utilizand
print('MSE: ', np.mean((least_square - b)**2))
```

```
MSE:  0.00696082344057169
```

In [10]:
```
#calculo do coeficiente de determinação sobre os próprios dados de treino uti
#sklearn
print('R^2: ', metrics.r2_score(b, least_square))
```

```
R^2:  0.9940259913481798
```

Note o alto grau de ajuste do modelo aos próprios dados de treino. Aplica-se, então, o modelo obtido acima aos dados de teste.

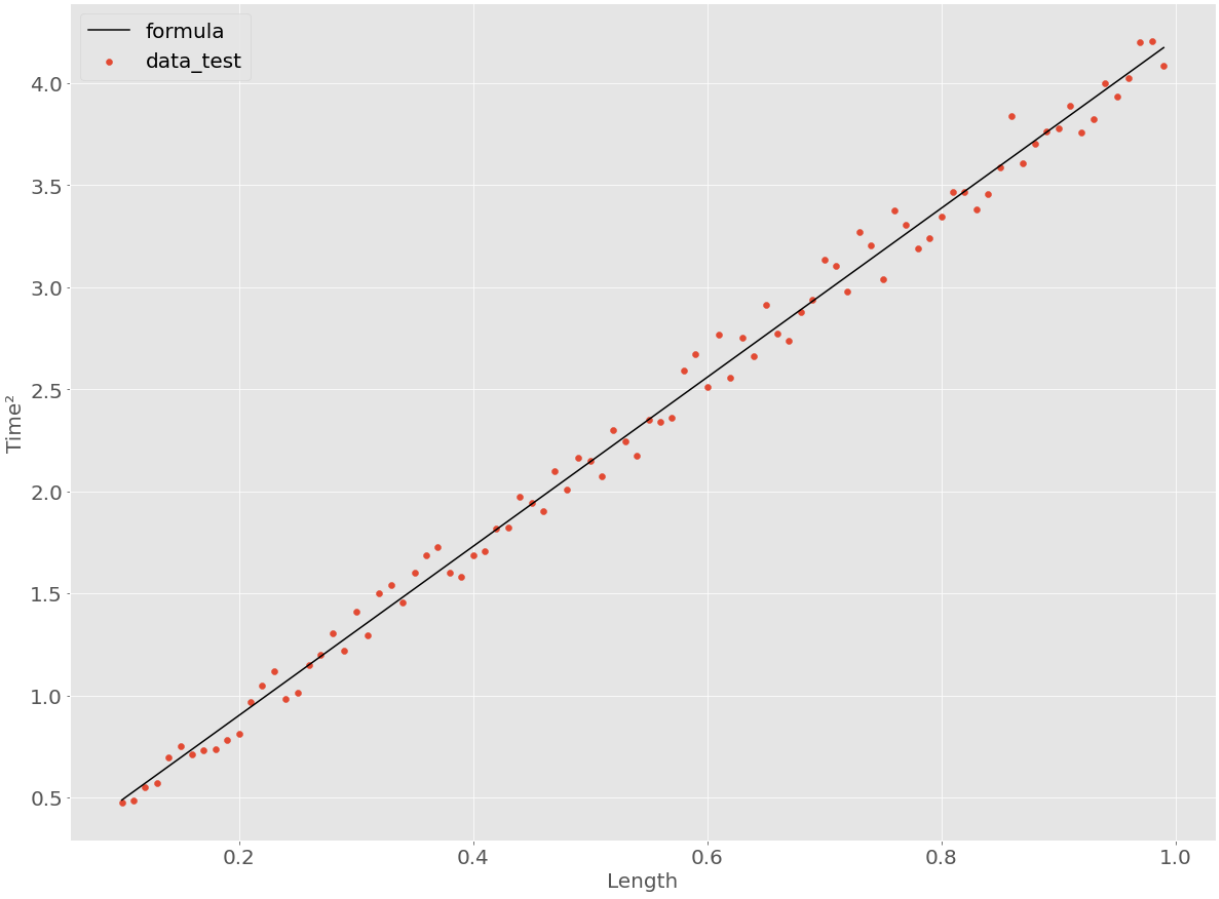
In [11]:
```
#por meio da função read_csv da biblioteca pandas, realizou-se a leitura da b
data_test = pd.read_csv('simple_pendulum_data_test.csv').dropna()

#print da descrição básica da base de dados utilizando o método describe da b
print(data_test.describe())

#armazenou-se em periodo_test os valores da coluna 'time' do dataframe utiliz
periodo_test = data_test['time'].values
```

```
#armazenou-se em comprimento_test os valores da coluna 'length' do dataframe
comprimento_test = data_test['length'].values
```

```
          length       time
count  90.000000  90.000000
mean    0.545000   1.478573
std     0.261247   0.382274
min     0.100000   0.690040
25%     0.322500   1.211450
50%     0.545000   1.523550
75%     0.767500   1.806550
max     0.990000   2.050300
```

In [12]:

```
#armazenou-se em t_test os valores do comprimento
t_test = comprimento_test
#armazenou-se em b_test os valores do periodo ao quadrado
b_test = (periodo_test) ** 2

#criação de uma figura com os dados de teste e a reta que minimiza o erro qua
#sendo no eixo X o comprimento e no eixo Y o periodo ao quadrado
fig, ax = plt.subplots()
plt.scatter(t_test, b_test, label='data_test')
plt.plot(t_linspace, least_square_plot, color='k', label='formula')
plt.xlabel('Length')
plt.ylabel('Time²')
plt.legend()
plt.show()
```



In [13]:

```
#calculo do erro quadrático médio sobre os dados de teste utilizando a funçã
print('MSE: ', np.mean((least_square - b_test)**2))
```

```
MSE:  0.007067851710805273
```

In [14]:
```
#calculo do coeficiente de determinação sobre os dados de teste utilizando a
print('R^2: ', metrics.r2_score(b_test, least_square))
```

```
R^2:  0.9939314951817307
```

Conclusões:

Note que o modelo linear treinado se mostrou excelente para os dados de teste. Era de se esperar esse resultado, visto que o modelo é previsto teoricamente, e o mesmo experimento de pêndulo simples realizado em locais diferentes da Terra apenas é diferenciado pela aceleração gravidade local, a qual não sofre diferenças significativas sobre os pontos da superfície terrestre.

A equação obtida para a reta ajustada aos dados de treino é aproximadamente:

$$T^2 = 4,14l + 0,07$$

Além disso, a partir do coeficiente de l na equação da reta ajustada, encontramos um valor para a aceleração da gravidade de $9,54m/s^2$, o que é bem próximo do valor de $g = 9,80665m/s^2$ medido a 45° de latitude ao nível do mar, sendo, portanto, uma boa estimativa para g no local do experimento correspondente aos dados de treino.

Questão 4

Vejamos, então, o pseudocódigo para treinamento do modelo de regressão linear por mínimos quadrados:

$\mathbf{(1)}$ Plote uma reta aleatória sobre o conjunto de dados de treino.
$\mathbf{(2)}$ Calcule a diferença ao quadrado entre a ordenada de cada dado e da reta (distância vertical).
$\mathbf{(3)}$ Faça uma média de todas as diferenças do passo anterior (erro quadrático médio).
$\mathbf{(4)}$ Repita os passos de 1 a 3 várias vezes e selecione a reta que forneceu o menor erro quadrático médio.
$\mathbf{(5)}$ Avalie a performance da reta ajustada sobre os dados de teste.

Questão 5

Podemos ver no código da questão 3 que as principais bibliotecas utilizadas de acordo com a linguagem Python para o uso do método de regressão são Pandas, NumPy, Matplotlib e Scikit-learn.

Pandas: Utilizada para ler e manipular bases de dados a serem analisados pelo método.
NumPy: Utilizada para manipular dados na forma matricial de maneira mais eficiente.
Matplotlib: Utilizada para visualização dos dados em análise.
Scikit-learn: Utilizada para calcular métricas e avaliar o modelo

Questão 6

O coeficiente de determinação, também chamado de R^2, é uma medida de ajuste de um modelo estatístico linear generalizado, como a regressão linear simples ou múltipla, aos valores observados de uma variável aleatória. O R^2 varia entre 0 e 1, por vezes sendo expresso em termos percentuais. Nesse caso, expressa a quantidade da variância dos dados que é

explicada pelo modelo linear. Assim, quanto maior o R^2, mais explicativo é o modelo linear, ou seja, melhor ele se ajusta à amostra.

Como visto na questão 3, o valor do R^2 pode ser calculado em Python através da função "sklearn.metrics.r2_score" .

A inclusão de inúmeras variáveis, mesmo que tenham muito pouco poder explicativo sobre a variável dependente, aumentarão o valor de R^2. Isto incentiva a inclusão indiscriminada de variáveis, prejudicando o princípio da parcimônia ("menos é melhor", para mais detalhes, ver https://pt.wikipedia.org/wiki/Parcim%C3%B4nia e https://pt.wikipedia.org/wiki/Navalha_de_Ockham). Para combater esta tendência, podemos usar uma medida alternativa do coeficiente de determinação, que penaliza a inclusão de variáveis independentes pouco explicativas. Trata-se do R^2 ajustado:

$$\bar{R^2} = 1 - \frac{n-1}{n-(k+1)}\left(1-R^2\right)$$

onde n é o tamanho do conjunto de dados e k representa o número de variáveis explicativas, excluindo o termo constante. Note que a inclusão de mais variáveis com pouco poder explicativo prejudica o valor do R^2 ajustado, porque aumenta k uma unidade, sem aumentar substancialmente o R^2.

Vamos provar que $\bar{R^2} \leq R^2$, estando de acordo com o princípio da parcimônia. Sendo $\alpha = \frac{n-1}{n-(k+1)}$, temos $\bar{R^2} = 1 - \alpha \cdot \left(1-R^2\right) = \alpha R^2 - \alpha + 1$. Como $k > 1$, é evidente que $\alpha > 1$, ou seja, $1 - \alpha < 0$. Temos:

$$R^2 \leq 1$$

$$(1-\alpha)R^2 \geq (1-\alpha)$$

$$R^2 - \alpha R^2 \geq 1 - \alpha$$

$$R^2 \geq \alpha R^2 - \alpha + 1$$

$$\bar{R^2} \leq R^2$$

Como queríamos demonstrar. Assim, o R^2 ajustado se mostra útil em regressões multivariadas, equilibrando acurácia com simplicidade.

Comentários gerais

O trabalho possibilitou que o grupo conhecesse profundamente um dos métodos de Ciência de Dados, incluindo a história do método, modelos frequentemente utilizados e uma aplicação. Além disso, discutiu-se as principais formas de comparação de performance de modelos.

O trabalho foi realizado por chamadas de vídeo. A dupla discutia qual a melhor maneira de responder cada uma das questões propostas. Desse modo, ambos os alunos sabiam e participavam do que estava sendo realizado.

Bibliografia

https://scikit-learn.org/stable/
https://pandas.pydata.org/
https://numpy.org/
https://matplotlib.org/
https://en.wikipedia.org/wiki/Linear_regression
https://pt.wikipedia.org/wiki/M%C3%A9todo_dos_m%C3%ADnimos_quadrados
https://scikit-learn.org/stable/modules/generated/sklearn.metrics.r2_score.html#sklearn.metrics.r2_score
https://en.wikipedia.org/wiki/Coefficient_of_determination
https://pt.wikipedia.org/wiki/Coeficiente_de_determina%C3%A7%C3%A3o
https://pt.wikipedia.org/wiki/Regress%C3%A3o_(estat%C3%ADstica)
https://pt.wikipedia.org/wiki/Regress%C3%A3o_linear
https://en.wikipedia.org/wiki/Linear_regression
https://www.ime.usp.br/~cgcap/map2210/NotasDeAulas/topico_7_MMQ.pdf
https://pt.wikipedia.org/wiki/Parcim%C3%B4nia
https://pt.wikipedia.org/wiki/Navalha_de_Ockham
https://en.wikipedia.org/wiki/Polynomial_regression
https://en.wikipedia.org/wiki/Gauss%E2%80%93Markov_theorem
https://rpubs.com/JulhinhaM/395633
https://en.wikipedia.org/wiki/Ridge_regression
https://en.wikipedia.org/wiki/Lasso_(statistics)