**Detecção de anomalias em despesas dos deputados estaduais**

**de São Paulo por meio de K-Means**

Rodolfo Orlando Viana¹\*; Ana Julia Righetto2

1 Avenida Maria Fernandes Cavallari, 3099 – Jardim Cavallari; 17526-341 Marília, São Paulo, Brasil

2 Nome da Empresa ou Instituição (opcional). Titulação ou função ou departamento. Endereço completo (pessoal ou profissional) – Bairro; 00000-000 Cidade, Estado, País

\*Rodolfo Orlando Viana: eu@rodolfoviana.com.br

**Detecção de anomalias em despesas dos deputados estaduais**

**de São Paulo por meio de K-Means**

**Resumo**

O presente trabalho investigou anomalias em gastos de fundos públicos recebidos pelos deputados da Assembleia Legislativa do Estado de São Paulo [Alesp] por meio da "verba de gabinete". Com as alocações de 2022 superando os anos anteriores e alegações de malversação de recuso público feitas por órgãos de controle, torna-se imperativo examinar esses gastos. Empregamos aprendizado de máquina não supervisionado, especificamente a clusterização K-Means com método de inicialização K-Means++, para discernir anomalias nas despesas. Embora não rotulemos conclusivamente as transações como fraudulentas, nossa metodologia oferece um arcabouço para identificar possíveis inconsistências financeiras, auxiliando órgãos de supervisão em suas análises.

**Palavras-chave:** Alesp; recursos públicos; clusterização; K-Means++; aprendizado de máquina não supervisionado.

**Anomaly detection in expenses of state deputies of São Paulo using K-Means**

**Abstract**

This study investigated anomalies in the spending of public funds received by deputies of the São Paulo State Legislative Assembly [Alesp] through the "office allowance". With the 2022 allocations surpassing previous years and allegations of misappropriation of public funds made by oversight bodies, it becomes imperative to examine these expenditures. We employed unsupervised machine learning, specifically the K-Means clustering with the K-Means++ initialization method, to discern anomalies in expenses. While we do not conclusively label transactions as fraudulent, our methodology provides a framework to identify potential financial inconsistencies, assisting oversight bodies in their reviews.

**Keywords:** Alesp; public funds; clustering; K-Means++; unsupervised machine learning

**Introdução**

Cada um dos 94 parlamentares da Assembleia Legislativa do Estado de São Paulo [Alesp] tem direito aos Auxílio-Encargos Gerais de Gabinete de Deputado e Auxílio-Hospedagem, referenciados conjuntamente como “verba de gabinete”. Tal direito foi conferido pela resolução 783, artigo 11, de 1º de julho de 1997. Trata-se de um valor mensal devido pelo Estado aos deputados a fim de que eles possam cobrir gastos com o funcionamento e manutenção dos gabinetes, com hospedagem e demais despesas inerentes ao pleno exercício das atividades parlamentares.

Tais gastos previstos na legislação são agregados em 11 categorias, dentre as quais materiais e serviços gráficos, consultoria, combustíveis, locação de automóveis, hospedagem. Em 2022, considerando resolução 783, de 1º de julho de 19971, que estipula o limite máximo da verba de gabinete em 1.250 unidades fiscais do Estado de São Paulo [Ufesp], e o valor da Ufesp em R$ 31,97, o limite mensal da verba de gabinete que poderia ser ressarcido por deputado no ano passado foi de R$ 39.962,50.

Naquele ano, o valor total empenhado para custeio da verba de gabinete perfez R$ 26.652.243,51. O montante foi 24,43% maior que a soma em 2021, de R$ 21.419.316,88, e menor do que o valor anotado na rubrica para 2023, de R$ 28.607.099,96. Caso este montante se cumpra neste ano, será a primeira vez que o valor ultrapassa R$ 28,5 milhões desde 2018.

Tais somas de recursos públicos podem servir, ainda que parcialmente, para infringir a lei. Um exemplo é o processo investigatório SEI 29.0001.0246360.2021-544, cujo pedido de instauração foi feito pelo Procurador Mario Antonio de Campos Tebet em 5 de maio de 2022. A peça elenca possível malversação no uso da verba de gabinete por parte do deputado estadual Murilo Felix, que a teria empregado para pagar pela locação de imóveis pertencentes a aliados políticos e nunca utilizados.

Com este contexto, o presente trabalho busca ser um instrumento para avaliação de malversação de dinheiro público por meio de aprendizado de máquina não supervisionado. O objetivo desta peça não é afirmar peremptoriamente se determinada despesa é fraudulenta ou não; seu escopo é servir de ferramenta para uma observação inicial dos gastos, que podem ser analisados por meio de clusterização, onde se objetiva encontrar um grupo de despesas cujos valores indicam possíveis anomalias.

**Material e Métodos**

A primeira etapa consistiu na captura, limpeza e normalização de dados relacionados às despesas dos deputados. Tais registros estão disponíveis no Portal de Dados Abertos da Alesp, e datam desde 2002. Inicialmente foram trabalhadas as despesas relacionadas a alimentação e hospedagem compreendidas entre os anos de 2018 e 2022. Dado o contexto temporal dos gastos, realizou-se a deflação dos valores até 31 de dezembro de 2022 seguindo o índice de preço ao consumidor amplo [IPCA], conforme divulgado pelo IBGE a partir de dados do Banco Central. Isso permitiu que os valores de todos os anos se mantivessem no mesmo contexto temporal.

Uma análise exploratória foi realizada para compreender os dados e sua dispersão no conjunto. No quinquênio observado, foram 14.127 registros de despesas em 4.414 números de CNPJ únicos, totalizando R$ 4.135.666,70. Cada despesa apresentou valor médio de R$ 292,75, porém com desvio-padrão elevado (R$ 681,29), indicando significativa dispersão dos dados em relação à média. O coeficiente de variação de 232,72% demonstrou alto grau de variabilidade relativo à média.

Notou-se ainda que a média é superior ao terceiro quartil. Isso indica que o conjunto de dados está inclinado para valores mais baixos, apesar da significante presença de outliers que puxa o terceiro quartil para cima. Graficamente, o valor médio maior que o terceiro quartil sugere assimetria positiva: a cauda do lado direito é mais longa do que do lado esquerdo. Essa indicação é corroborada com a assimetria de 7, enquanto a curtose de 64,8 comprova cauda longa e picos acentuados em comparação à distribuição normal.

Tabela 1. Estatísticas dos dados analisados

|  |  |
| --- | --- |
| Medida | Valor |
| Contagem | 14.127 |
| Média | R$ 292,749112 |
| Desvio-padrão | R$ 681,290247 |
| Mínimo | R$ 0 |
| 1º Quartil | R$ 54,265 |
| 2º Quartil | R$ 115,33 |
| 3º Quartil | R$ 253,33 |
| Máximo | R$ 10.259,41 |
| Coeficiente de variação | 232,72154207553792%­­ |
| ­­Assimetria | 7,040865107241919 |
| Curtose | 64,79066970927987 |

Fonte: Dados originais da pesquisa

Em seguida, foi construído um algoritmo de clusterização por K-Means. A organização dos conjuntos é feita com a determinação aleatória de um centroide, um ponto que observa a distância euclidiana dos demais dados em relação a ele. Para dados univariados como os trabalhados aqui, a distância euclidiana é calculada com

sendo x um ponto de dados e y o ponto escolhido como centroide.

Dado um conjunto de observações , o algoritmo de K-Means reparte as n observações em conjuntos a fim de minimizar a soma dos quadrados dentro do cluster. Seu objetivo é encontrar

sendo a média dos pontos em .

Para se obter a quantidade ideal de clusters, é utilizado o método Silhouette, que mede a coesão entre um ponto de dado e seu cluster, medida obtida por

onde a(i) é a distância média entre ponto i e os outros pontos no mesmo cluster, e b(i) é a menor distância média entre ponto i e pontos em clusters diferentes.

Neste trabalho se utilizou o método de inicialização K-Means++, em que o centroide atual é escolhido com base na distância entre os pontos de dados e o centroide anteriormente selecionado. Essa iteração reduz as chances de o algoritmo K-Means convergir para uma solução abaixo do ideal.

Dado um conjunto de pontos D e um conjunto de centroides já selecionados C, a probabilidade de se escolher o ponto de dado x como próximo centroide é calculada por meio de

sendo a distância entre o ponto e o centroide mais próximo em .

Com os centroides inicializados, cada ponto é atribuído ao centroide mais próximo. Tais pontos de dados próximos ao centroide formam clusters. Considerando o ponto x e um conjunto de centroides C, o rótulo do cluster l ao qual x pertence é computado por

Em seguida, cada centroide é recalculado tomando a média da distância de todos os pontos a eles atribuídos,

onde é o conjunto de todos os pontos de dados atribuídos ao centroide i.

A cada iteração de atualização de centroides é computado o custo da função — a inércia. Para conjunto univariado, a operação segue

onde é o centroide do cluster para o qual o ponto foi atribuído.

O algoritmo desenvolvido também adota critérios de convergência avançados ao comparar o movimento dos centroides entre iterações. Sendo o conjunto de centroides na iteração , o algoritmo converge se

onde tol é a tolerância especificada, e denota a distância euclidiana.

Aplicada aos dados de despesas dos deputados estaduais, esta metodologia permitiu agrupar aquelas que, segundo seus valores, requerem investigação mais apurada por parte dos órgãos de controle.

**Resultados e Discussão**

O título da seção Resultados e Discussão deve ser alinhado à esquerda, grafado em negrito com as primeiras letras das palavras em letras maiúsculas. É permitido que a seção seja dividida em subtópicos com formatação de acordo com a descrição no item 1.1 Formato e margens, apresentados na mesma ordem da seção Material e Métodos. Nesta seção devem ser apresentados, discutidos e interpretados os resultados obtidos no trabalho, ou seja, autores devem fazer uma discussão comparativa dos resultados do seu trabalho com aqueles existentes na literatura e elaborar uma análise crítica dos dados, destacando as limitações e pontos positivos dos resultados.

**Considerações Finais**

O título da seção Conclusão(ões) ou Considerações Finais deve ser alinhado à esquerda e grafado em negrito. Fica a critério do aluno e do orientador a escolha de qual termo melhor se adequa ao trabalho. Esta seção deve conter frases curtas, apresentando as conclusões e inferências elaboradas a partir da discussão dos resultados. É importante que estas frases não sejam meras reproduções dos resultados, respondendo aos objetivos propostos no trabalho. Os autores não devem, em hipótese alguma, mencionar, citar ou reproduzir resultados de outros estudos na(s) conclusão(ões) ou considerações finais do TCC. Por fim, salienta-se que essa seção não deve conter tabelas ou figuras, sendo redigida de forma sucinta.

**Agradecimento**

Agradeço a Pedro Orlando, meu afilhado de seis anos que, com seus convites para assistir a vídeos do Enaldinho ou “fazer um piquenique” na sala de casa, conseguiu me distrair deste trabalho ao tempo que recarregou minhas energias para nele prosseguir.

**Referências**

Arthur, D.; Vassilvitskii, S. 2007. K-Means++: The advantages of careful seeding. Proceedings of Annual ACM-SIAM Symposium on Discrete Algorithms: 1027-1035.

Assembleia Legislativa do Estado de São Paulo [Alesp]. 1997. Resolução n. 783, de 1° de julho de 1997. Altera a Resolução n° 776, de 14/10/1996, que implantou a nova estrutura administrativa, cria o Núcleo de Qualidade e institui a verba de gabinete. Disponível em: https://www.al.sp.gov.br/repositorio/legislacao/resolucao.alesp/1997/original-resolucao.alesp-783-01.07.1997.html. Acesso em: 19 março 2023.

Hartigan, J.A.; Wong, M.A. 1979. Algorithm AS 136: A K-Means clustering algorithm. Journal of the Royal Statistical Society, Series C (Applied Statistics) 28: 100-08.

Lloyd, S. 1982. Least squares quantization in PCM. IEEE Transactions on Information Theory 28: 129-137.

MacQueen, J. 1967. Classification and analysis of multivariate observations. In: 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967, Los Angeles, LA, Estados Unidos, Anais… p. 281-297.

Ministério Público de São Paulo. 2022. Sistema Eletrônico de Informações. Disponível em: https://www.mpsp.mp.br/sei-sistema-eletronico-de-informacoes Acesso em: 26 março 2023.

Morissette, L.; Chartier, S. 2013. The K-Means clustering technique: General considerations and implementation in Mathematica. Tutorials in Quantitative Methods for Psychology 9: 15-24.

Secretaria da Fazenda e Planejamento do Governo do Estado de São Paulo. 2023. Execução orçamentária e financeira. Disponível em: https://www.fazenda.sp.gov.br/sigeolei131/paginas/flexconsdespesa.aspx. Acesso em: 19 março 2023.

Secretaria da Fazenda e Planejamento do Governo do Estado de São Paulo. 2023. Índices. Disponível em: https://portal.fazenda.sp.gov.br/Paginas/Indices.aspx. Acesso em: 26 março 2023.

**Apêndice**

1. **Código-fonte do algoritmo, comentado**

from typing import Tuple

import numpy as np

class KMeans:

"""

k-means com critérios de convergência aprimorados.

Atributos:

k (int): Número de clusters.

max\_iters (int): Número máximo de iterações para o k-means.

tol (float): Tolerância de convergência baseada no movimento do centroide.

n\_init (int): Número de vezes que o algoritmo será executado com diferentes seeds de centroides.

threshold (int): Percentil para detecção de anomalias.

centroids (np.ndarray): Centroides para os clusters.

"""

def \_\_init\_\_(

self,

k: int = 2,

max\_iters: int = 100,

tol: float = 1e-4,

n\_init: int = 30,

threshold: int = 95,

centroids: np.ndarray = None,

):

"""

Inicialização com parâmetros especificados.

"""

self.k = k

self.max\_iters = max\_iters

self.tol = tol

self.n\_init = n\_init

self.threshold = threshold

self.centroids = centroids

@staticmethod

def \_kpp\_init(data: np.ndarray, k: int) -> np.ndarray:

"""

Inicializa os centroides usando o método k-means++.

Argumentos:

data (np.ndarray): Dados de entrada.

k (int): Número de centroides desejados.

Retorna:

centroids (np.ndarray): Centroides inicializados.

"""

centroids = [data[np.random.choice(len(data))]]

for \_ in range(1, k):

squared\_dist = np.array(

[np.min([

np.linalg.norm(c - x) \*\* 2 for c in centroids

]) for x in data]

)

probs = squared\_dist / squared\_dist.sum()

centroid = data[np.argmax(probs)]

centroids.append(centroid)

return np.array(centroids)

def get\_optimal\_k(self, data: np.ndarray, k\_max: int = 10) -> int:

"""

Determina o número ideal de clusters k com o método Silhouette.

Argumentos:

data (np.ndarray): Dados sobre os quais o número ideal de k será determinado.

k\_max (int, opcional): Valor máximo de k. Valor padrão: 10.

Retorna:

optimal\_k (int): Número ideal de clusters.

"""

max\_silhouette = -1

optimal\_k = 2

def get\_score(data, labels):

"""

Calcula a média de Silhouette score dadas as labels de clusterização do conjunto de dados.

Argumentos:

data (np.ndarray): Dados de entrada.

labels (np.ndarray): Labels da clusterização para os pontos de dados.

Retorna:

(float): Silhouette score médio.

"""

unique\_labels = np.unique(labels)

silhouettes = []

for i, label in enumerate(labels):

points\_within\_cluster = data[labels == label]

avg\_dist\_within\_cluster = np.mean(

np.linalg.norm(

points\_within\_cluster - data[i],

axis=1)

)

min\_avg\_dists = [

np.mean(

np.linalg.norm(

data[labels == other\_label] - data[i],

axis=1)

)

for other\_label in unique\_labels

if other\_label != label

]

silhouette\_value = (

np.min(min\_avg\_dists) - avg\_dist\_within\_cluster

) / max(avg\_dist\_within\_cluster, np.min(min\_avg\_dists))

silhouettes.append(silhouette\_value)

return np.mean(silhouettes)

for k in range(2, k\_max + 1):

self.k = k

self.fit(data)

silhouette\_avg = get\_score(data, self.labels)

if silhouette\_avg > max\_silhouette:

max\_silhouette = silhouette\_avg

optimal\_k = k

return optimal\_k

def \_single\_run(self, data: np.ndarray) -> Tuple[np.ndarray, np.ndarray, float]:

"""

Realiza execução única do algoritmo k-means.

Argumentos:

data (np.ndarray): Dados de entrada.

Retorna:

centroids (np.ndarray): Melhores centroides após a execução do k-means.

labels (np.ndarray): Atribuições de cluster para cada ponto de dado.

inertia (float): Distância total dos pontos de dados a partir de seus

centroides atribuídos.

"""

centroids = self.\_kpp\_init(data, self.k)

for \_ in range(self.max\_iters):

dist = np.linalg.norm(data[:, np.newaxis] - centroids, axis=2)

labels = np.argmin(dist, axis=1)

new\_centroids = np.array(

[data[labels == i].mean(axis=0) for i in range(self.k)]

)

if np.all(np.abs(new\_centroids - centroids) < self.tol):

break

centroids = new\_centroids

inertia = np.sum(

[

np.linalg.norm(data[i] - centroids[labels[i]]) \*\* 2

for i in range(len(data))

]

)

return centroids, labels, inertia

def fit(self, data: np.ndarray) -> None:

"""

Ajusta o algoritmo k-means aos dados.

Argumento:

data (np.ndarray): Dados de entrada.

"""

min\_inertia = float("inf")

best\_centroids = None

best\_labels = None

for \_ in range(self.n\_init):

centroids, labels, inertia = self.\_single\_run(data)

if inertia < min\_inertia:

min\_inertia = inertia

best\_centroids = centroids

best\_labels = labels

self.centroids = best\_centroids

self.labels = best\_labels

def detect(self, data: np.ndarray) -> np.ndarray:

"""

Detecta anomalias nos dados com base na distância ao centroide mais próximo.

Argumentos:

data (np.ndarray): Dados de entrada.

Retorna:

anomalies (np.ndarray): Anomalias detectadas.

"""

dist = np.min(

np.linalg.norm(data[:, np.newaxis] - self.centroids, axis=2), axis=1

)

threshold = np.percentile(dist, self.threshold)

anomalies = data[dist > threshold]

return anomalies