**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 19971. 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$ 31,972, 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$ 26.652.243,513. 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 são indicativos de 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.

Uma análise exploratória foi realizada para compreender a dispersão dos dados no conjunto. Em seguida, foi construído um algoritmo de clusterização por meio da classificação por K-Means.

Na clusterização por K-Means, a coleção de dados é atribuída a conjuntos distintos com base em sua similaridade. 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. Aqueles mais próximos ao centroide formam um conjunto. Em seguida, a localização do centroide é recalculada considerando o custo da função, e novamente os pontos mais próximos são agregados em conjuntos. Isso ocorre repetidas vezes até se obter convergência entre centroide e dados. Em outros termos, a classificação por K-Means particiona uma população de dimensões em conjuntos com base em uma amostra.

Para a maioria dos usos de K-Means, o algoritmo de Lloyd é escolhido, motivo pelo qual também é utilizado neste trabalho. Sabe-se que este algoritmo pode ser fortemente influenciado pela seleção dos centroides iniciais, levando à convergência a um optimum local7, o que pode gerar clusters sem critérios rígidos. Para isso, o algoritmo Hartigan-Wong poderia ser um substitutivo. Contudo, análises iniciais mostraram que, neste trabalho, Hartigan-Wong não apresenta resultados diferentes de LLoyd.

Neste trabalho também se utilizou o método de inicialização K-Means++. Sua tendência é espalhar os centroides iniciais pelos dados, reduzindo as chances de o algoritmo K-Means convergir para uma solução abaixo do ideal — ou seja, sua abordagem garante que os pontos mais distantes dos centroides existentes tenham maior probabilidade de serem escolhidos como novos centroides.

A inicialização por K-Means++ segue os seguintes passos:

1. Seleção aleatória do primeiro centroide;
2. Cálculo de , as distâncias entre pontos de dados e centro mais próximo;
3. Escolha aleatória de um novo ponto de dados como um novo centro, usando uma distribuição de probabilidade ponderada onde um ponto é escolhido com probabilidade proporcional a — isto é, tendo um conjunto de pontos e um conjunto de centroides já inicializados, para qualquer ponto em , a probabilidade de escolher como o próximo centroide é , onde é a distância de a partir do centroide mais próximo em ;
4. Repetição dos passos 2 e 3 até que os centros sejam escolhidos.

Aplicada aos dados de despesas dos deputados estaduais, esta metodologia permitiu agrupar aquelas que, segundo o valor, 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**

from typing import Tuple

import numpy as np

class KMeans:

"""

k-means clustering class with enhanced convergence criteria.

Attributes:

k (int): Number of clusters.

max\_iters (int): Maximum number of iterations for k-means.

tol (float): Convergence tolerance based on centroid movement.

n\_init (int): Number of times the algorithm will be run with different centroid seeds.

threshold (int): Percentile for anomaly detection.

centroids (np.ndarray): Centroids for the 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

):

"""

Initialize k-means with specified parameters.

"""

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:

"""

Initialize the centroids using the k-means++ method.

Args:

data (np.ndarray): Input data.

k (int): Number of desired centroids.

Returns:

centroids (np.ndarray): Initialized centroids.

"""

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

"""

Perform a single run of the k-means algorithm.

Args:

data (np.ndarray): Input data.

Returns:

centroids (np.ndarray): Best centroids after running k-means.

labels (np.ndarray): Cluster assignments for each data point.

inertia (float): Total distance of data points from their assigned centroids.

"""

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)])

# Check for convergence

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

break

centroids = new\_centroids

# Calculate inertia (sum of squared distances to the nearest centroid)

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:

"""

Fit the k-means algorithm to the data.

Args:

data (np.ndarray): Input data.

"""

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:

"""

Detect anomalies in the data based on the distance to the nearest centroid.

Args:

data (np.ndarray): Input data.

Returns:

anomalies (np.ndarray): Detected anomalies.

"""

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