Aldo Daniel Villaseñor Fierro

A01637907

Promedios móviles

In [ ]:
```
import numpy as np
#17     21      19      23      18      16      20      18      22      20      15
gal=[17,21,19,23,18,16,20,18,22,20,15,22]
semana=np.arange(1,13)
# Función que calcula la media móvil de una secuencia
def meanmov(a, n):
    #Utiliza la función cumsum para calcular la suma acumulada de los valores de la se
    ret = np.cumsum(a, dtype=float)
    ret[n:] = ret[n:] - ret[:-n]
    return ret[n - 1:] / n

#Promedios moviles
mov=meanmov(gal,3)

error=np.mean((mov-gal[2:])**2)

print("El error cuadrático medio es: ",error)
```

```
El error cuadrático medio es:  5.0
```

In [ ]:
```
print("Datos recopilados",gal)
print("Resultado de la media movil",mov)
```

```
Datos recopilados [17, 21, 19, 23, 18, 16, 20, 18, 22, 20, 15, 22]
Resultado de la media movil [19. 21. 20. 19. 18. 18. 20. 20. 19. 19.]
```

In [ ]:
```
import plotly.express as px

fig = px.line(x=semana, y=gal)
fig.add_scatter(x=semana[3:], y=mov, mode='lines', name='Promedio movil')
fig.show()
```

Promedios móviles ponderados

In [ ]:
```
n = len(gal)
p = [0] * n
e = [0] * n

for i in range(n - 3):
    p[i] = (1/6) * gal[i] + (2/6) * gal[i + 1] + (3/6) * gal[i + 2]

p=np.array(p)
gal_array=np.array(gal)
error=np.mean((gal_array[3:]-p[:-3])**2)
print("El error cuadrático medio es: ",error)
```

```
El error cuadrático medio es:  11.49074074074074
```

In [ ]:
```
print("Datos recopilados",gal)
print("Resultado de la media movil ponderada",p)
```

```
Datos recopilados [17, 21, 19, 23, 18, 16, 20, 18, 22, 20, 15, 22]
Resultado de la media movil ponderada [19.33333333 21.33333333 19.83333333 17.8333333
3 18.33333333 18.33333333
 20.33333333 20.33333333 17.83333333  0.          0.          0.        ]
```

In [ ]:
```
fig = px.line(x=semana, y=gal)
fig.add_scatter(x=semana[3:], y=p, mode='lines', name='Promedio movil')
fig.show()
```

Suavizado exponencial

In [ ]:
```
n = len(gal)
p = [None] * n
e = [None] * n
p[0] = gal[0]
a = 0.2

for i in range(1, n):
    p[i] = a * gal[i - 1] + (1 - a) * p[i - 1]

p=np.array(p)
gal_array=np.array(gal)
error=np.mean((gal_array[1:]-p[1:])**2)
print("El error cuadrático medio es: ",error)
```

```
El error cuadrático medio es:  8.982230675137561
```

In [ ]:
```
print("Datos recopilados",gal)
print("Resultado de suavizado exponencial",p)
```

```
Datos recopilados [17, 21, 19, 23, 18, 16, 20, 18, 22, 20, 15, 22]
Resultado de suavizado exponencial [17.         17.         17.8        18.04       1
9.032      18.8256
 18.26048    18.608384   18.4867072  19.18936576 19.35149261 18.48119409]
```

In [ ]:
```
fig = px.line(x=semana, y=gal)
fig.add_scatter(x=semana[1:], y=p[1:], mode='lines', name='Promedio movil')
fig.show()
```

Optimización de suavizado exponencial

In [ ]:
```
# Función para calcular el CME dado un valor de a
def calculate_CME(a, y):
    n = len(y)
    p = [None] * n
    e = [None] * n
    p[0] = y[0]
    a = 0.2

    for i in range(1, n):
        p[i] = a * y[i - 1] + (1 - a) * p[i - 1]
```

```
    p=np.array(p)
    y_ar=np.array(y)
    error=np.mean((y_ar[1:]-p[1:])**2)
    return error

# Valores de "a" a evaluar
a_values = np.linspace(0.01, 1.0, 1000)

# Inicializa el CME mínimo y el valor de "a" correspondiente
min_CME = float('inf')
best_a = None

# Encuentra el valor de "a" que minimiza el CME
for a in a_values:
    current_CME = calculate_CME(a, gal)
    if current_CME < min_CME:
        min_CME = current_CME
        best_a = a

print("El mejor valor de 'a' es:", best_a)
print("CME mínimo correspondiente:", min_CME)
```

```
El mejor valor de 'a' es: 0.01
CME mínimo correspondiente: 8.982230675137561
```

Problema 2

Se registró el precio de las acciones de una compañía al cierre de cada día hábil del 24 de agosto al 16 de septiembre. Los datos recopilados son:

a. Use un promedio móvil de tres días para suavizar la serie de tiempo y pronosticar las ventas para el día 19 de septiembre.

In [ ]:
```
t = np.arange(1, 18)
y = np.array([81.32, 81.10, 80.38, 81.34, 80.54, 80.62, 79.54, 79.46, 81.02, 80.98, 80

def meanmov(a, n):
    ret = np.cumsum(a, dtype=float)
    ret[n:] = ret[n:] - ret[:-n]
    return ret[n - 1:] / n

#Promedios moviles
mov=meanmov(y,3)

error=np.mean((mov-y[2:])**2)

print("El pronostico para el día 19 de Septiembre es: ",mov[-1])

print("El error cuadrático medio es: ",error)
```

```
El pronostico para el día 19 de Septiembre es:  80.27333333333331
El error cuadrático medio es:  0.2171962962962983
```

In [ ]:
```
fig = px.line(x=t, y=y)
fig.add_scatter(x=t[3:], y=mov, mode='lines', name='Promedio movil')
fig.show()
```

b. Use un suavizado exponencial para suavizar la serie de tiempo y pronosticar las ventas para el día 19 de septiembre.

In [ ]:
```
t = np.arange(1, 18)
y = np.array([81.32, 81.10, 80.38, 81.34, 80.54, 80.62, 79.54, 79.46, 81.02, 80.98, 80
n = len(y)
p = [None] * n
e = [None] * n
p[0] = y[0]
a = 0.2

for i in range(1, n):
    p[i] = a * y[i - 1] + (1 - a) * p[i - 1]


p=np.array(p)
y_array=np.array(y)
error=np.mean((y_array[1:]-p[1:])**2)

p_len=len(p)

pred = a * y[ p_len - 1] + (1 - a) * p[p_len - 1]

print("El pronostico para el día 19 de Septiembre es: ",pred)

print("El error cuadrático medio es: ",error)
```

```
El pronostico para el día 19 de Septiembre es:  80.59539669276278
El error cuadrático medio es:  0.467199372704468
```

In [ ]:
```
fig = px.line(x=t, y=y)
fig.add_scatter(x=t[1:], y=p[1:], mode='lines', name='Promedio movil')
fig.show()
```

c. Preferencia entre los dos métodos:

En este caso, el método de promedios móviles es mejor que el método de suavizamiento exponencial, ya que el primero se ajusta mejor a los datos y no tiene un error tan grande como el segundo.