accenture

1) T_c: tiempo de vida de un componente en dias

$T_c \sim \mathcal{E}\left(\lambda = \frac{1}{10}\right)$ $\quad E(T) = 10 \quad$ $Var(T) = 100$

Probabilidad de quedar en falta < 0,05 en viaje de 365 dias
ΣT_c de los componentes > 365

$$\sum_n T_c = \sum_n \mathcal{E}\left(\lambda = \frac{1}{10}\right) = \Gamma\left(n, \frac{1}{10}\right)$$

Probabilidad $\Sigma T_c < 365$ tiene que ser menor a 0,05

$$P(\Sigma T_c < 365) < 0{,}05$$

$$n.T_c(\lambda) = \frac{\lambda^n}{\Gamma(n)} \cdot x^{n-1} \cdot e^{-\lambda x} = \frac{\lambda^n}{(n-1)!} \cdot x^{n-1} \cdot e^{-\lambda x}$$

Si $f(x) = \frac{dF(x)}{dx} \Rightarrow F(x) = \int F(x)\,dx = P(X < x)$

$$P(X < 365) = \int_0^{365} \frac{\left(\frac{1}{10}\right)^n}{(n-1)!} \cdot x^{(n-1)} \cdot e^{-\frac{1}{10} \cdot x}\, dx$$

Caso $n = 47$

$$P(x < 365) = \int_0^{365} \frac{0{,}1^{47}}{46!} \cdot x^{46} \cdot e^{-0{,}1x}\, dx = 0{,}0533 > 0{,}05 \quad \times$$

Caso $n = 48$

$$P(x < 365) = \int_0^{365} \frac{0{,}1^{48}}{47!} \cdot x^{47} \cdot e^{-0{,}1}\, dx = 0{,}03873 < 0{,}05 \quad \checkmark$$

Se necesitan 48 componentes al menos, para que la probabilidad de quedar en falta sea menor a 0,05

EJERCICIO 2

In [18]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

In [19]:
```
!wget "https://raw.githubusercontent.com/FIUBA-Posgrado-Inteligencia-Artificial/CE
```

```
--2022-08-16 03:31:09--  https://raw.githubusercontent.com/FIUBA-Posgrado-Intelige
ncia-Artificial/CEIA_ProbayEstadistica/master/Examen/temp_min_promedio.csv
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.13
3, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.13
3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1181 (1.2K) [text/plain]
Saving to: 'temp_min_promedio.csv.1'

temp_min_promedio.c 100%[===================>]   1.15K  --.-KB/s    in 0s

2022-08-16 03:31:09 (42.4 MB/s) - 'temp_min_promedio.csv.1' saved [1181/1181]
```

In [20]:
```
file="temp_min_promedio.csv"
temp=pd.read_csv("temp_min_promedio.csv")
```

In [21]:
```
temp.head()
```

Out[21]:

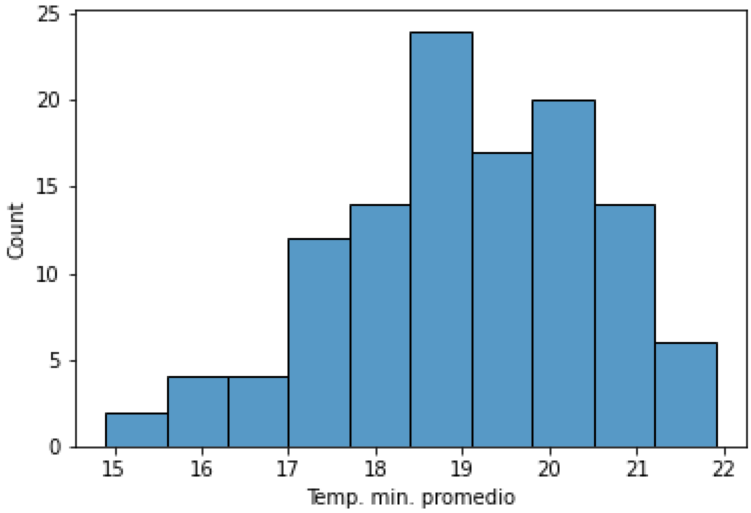
	Año	Temp. min. promedio
0	1906	17.6
1	1907	17.2
2	1908	15.5
3	1909	17.4
4	1910	17.4

In [22]:
```
muestras=len(temp["Temp. min. promedio"])
max=np.max(temp["Temp. min. promedio"])
min=np.min(temp["Temp. min. promedio"])
delta=(max-min)/10
delta,max,min
```

Out[22]:
```
(0.6999999999999998, 21.9, 14.9)
```

In [23]:
```
sns.histplot(temp,x="Temp. min. promedio", bins=10)
```

Out[23]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8dab90f10>
```



h9h10 es la cantidad de muestras con T>20.5

h10 es la cantidad de muestras con T>21.2

h9 es la cantidad de muestras entre 20.5<T<21.2

In [24]:
```
h9h10=np.count_nonzero(temp["Temp. min. promedio"] >= max-2*delta)
h10=np.count_nonzero(temp["Temp. min. promedio"] >= max-delta)
h9=h9h10-h10
```

Al quererse parecer a una función de densidad, calculamos el área bajo los bines involucrados en nuestra probabilidad de que la T sea mayor a 20.7°.

Esto nos da una probabilidad de 0.0957 de que la temperatura del siguiente enero sea mayor a 20.7°

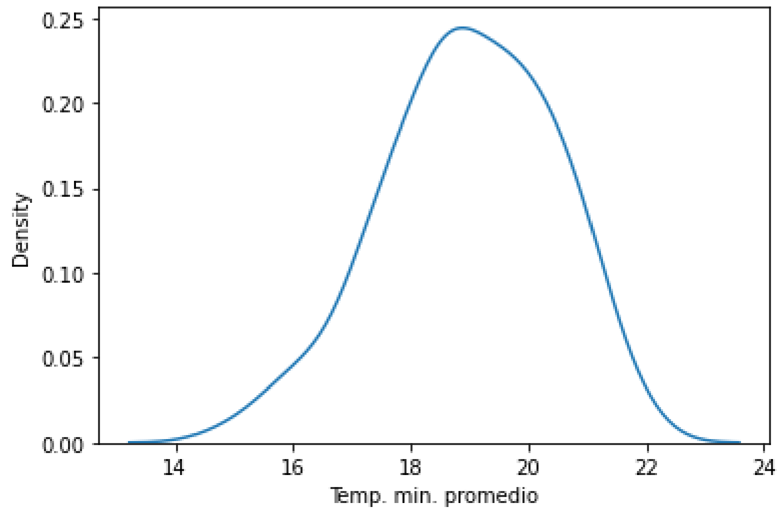
In [25]:
```
Tbuscada=20.7
Prob=((max-delta-Tbuscada)*h9+h10*delta)/muestras
Prob
```

Out[25]:
```
0.09572649572649572
```

Ploteando la funcion de densidad estimada por kernel, podemos ver que se ajusta al histograma, siendo esta continua, y asi reduciendo el error de varianza y sesgo

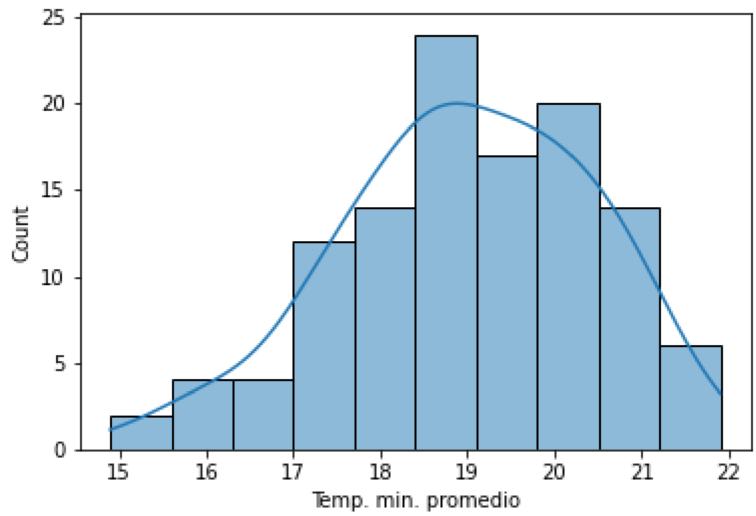
In [26]:
```
sns.kdeplot(temp['Temp. min. promedio'])
```

Out[26]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe8daa2c4d0>
```



In [27]:
```
sns.histplot(temp,x="Temp. min. promedio", bins=10,kde=True)
```

Out[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe8da9c1c10>



EJERCICIO 4

X: Dilatacion de rieles por temperatura

$X \sim \mathcal{N}(\mu, \sigma^2)$

In [28]:
```
import numpy as np
```

Necesitamos un pivote para determinar el IC. Para esto estimamos μ y σ^2 que al ser estimado obtenemos S^2

In [29]:
```
x=np.array([0.008, 0.032, 0.023, 0.015,0.022, 0.004, 0.028, 0.042, 0.06, 0.018])
n=len(x)
x_est=x.mean()
#sig2_est=np.power(x.std(),2)

sig2_est=np.sum(np.power(x-x_est,2))/(n-1)

x_piv=x_est
#Varianza del estimador es la varianza de la sumatoria de xi dividido n,
#sacando n fuera de la varianza sale al cuadrado, n^2
#la varianza de una suma de variables aleatorias independientes es n.Var(xi)
```

```
#Por lo tanto:
var_piv=sig2_est/n
#si evaluamos la bondad del estimador con n en el infinito, el la varianza es 0,
#asique se puede considerar a x_piv un estimador consistente
```

Nuestro pivote distribuye como una t de student

$(X-\mu)/S*\sqrt(n) \sim t(n-1)$ (parametros estimados, distribucion aproximada)

Buscamos el IC = [a<U<b], para esto estandarizamos la distribucion y buscamos los cuantiles 0.025 y 0.975 quedando en el medio nuestro IC de confianza de 0.95

$(X-\mu)/S*\sqrt(n) \sim t(n-1)$ (Xraya)

In [30]:
```
import scipy.stats as stats
a=stats.t(9).ppf(0.025)
b=stats.t(9).ppf(0.975)
a,b
```

Out[30]:
```
(-2.262157162740992, 2.2621571627409915)
```

Destandaricemos:

In [31]:
```
#x_estand=(x-x_piv)/np.sqrt(var_piv)
a_f=round(a*np.sqrt(var_piv)+x_piv,5)
b_f=round(b*np.sqrt(var_piv)+x_piv,5)
a_f,b_f
```

Out[31]:
```
(0.01336, 0.03704)
```

Finalmente nuestro IC(0.95)=(0.01336, 0.03704) en metros

Esto quiere decir que tenemos un 95% de certeza de que la media de dilatacion esta en el intervalo.

Si la media de la distribucion es mayor a 0.02 metros, se deben revisar las instalaciones debido al espacio ya establecido entre riel y riel. ¿Puede asegurar con un nivel de significacion de 0.05 que es necesario revisar las instalaciones? ¿Que error se podría estar cometiendo al tomar esta decision?

Planteo un test de hipotesis:

H0:No es necesario revisar las instalaciones ($\mu = 0.02$)

H1:Es necesario revisar las instalaciones ($\mu > 0.02$)

Nuestra funcion δ(x) para rechazar H0 sera 1 cuando el estadistico de prueba sea mayor a nuestro $k(1-\alpha)$ y 0 en otro caso.

Propongo el estadistico de prueba x estimado

In [32]:
```
x_est
```

Out[32]:
```
0.0252
```

Obtengo el cuantil de (1-0.05) de la distribucion t de student (n-1)

In [52]:
```
stats.t(9).ppf(0.95)
```

Out[52]:
```
1.8331129326536335
```

Estandarizo mi estadistico de prueba

In [43]:
```
(x_est-0.02)/np.sqrt(var_piv)
```

Out[43]:
```
0.9938926996864472
```

Como mi estadistico de prueba es menor que el cuantil de la t de student en 0.95, no rechazo H0, es decir que "no es necesario revisar las instalaciones", pudiendo cometer un error tipo II, que la media si sea mayor a 0.02 y que si haya sido necesario revisar las instalaciones.