In [15]:

```python
def subset_sum(numbers, target, partial=[]):
    s = sum(partial)

    # check if the partial sum is equals to target
    if s == target:

        Subsets.append(partial)

    if s >= target:
        return  # if we reach the number why bother to continue

    for i in range(len(numbers)):
        n = numbers[i]
        subset_sum(numbers, target, partial + [n])
```

In [16]:

```python
Subsets = []
subset_sum([f for f in range(3,37)],36)
```

In [17]:

```python
wsumL = []
for subset in Subsets:
    wsum = 0
    for t in range(len(subset)):
        if t == 0:
            wsum += 10*subset[0]-10
        else:
            wsum += (subset[t-1])*(subset[t]-1)
    wsumL.append(wsum+subset[-1])
```

In [18]:

```python
max(wsumL)
```

Out[18]:

```
510
```

In [ ]:

In [ ]:

In [2]:
```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [58]:
```python
def k_nbor(df, k, x):

    minIndex = []
    Dist = np.sum((df[:,0:-1] - x)**2, axis=1)
    minL = np.sort(Dist)[0:k]
    for k in range(k):
        minIndex.append(np.where(Dist == minL[k])[0][0])



    return np.sign(np.sum(df[minIndex,-1]))
```

In [62]:
```python
def uniform(df, gamma, x):
    return np.sign(np.sum(df[:,-1]*np.exp(-gamma*np.sum((df[:,:-1]-x)**2,axis=1))))
```

In [119]:
```python
def err(dftrain, dftest, parameter, method):
    ypred = []
    for t in dftest:
        ypred.append(method(dftrain, parameter, t[0:-1]))
    return np.sum((np.array(ypred) != dftest[:,-1]))/len(dftest)
```

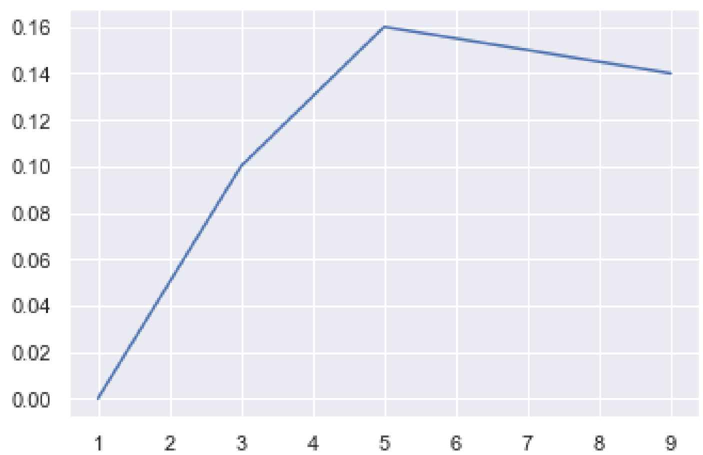
In [114]:
```python
dftrain = np.loadtxt('hw4_train.dat.txt')
```

In [115]:
```python
dftest = np.loadtxt('hw4_test.dat.txt')
```

In [121]:
```python
#Q11
K = [1,3,5,7,9]
E = []
for k in K:
    E.append(err(dftrain, dftrain, k, k_nbor))

sns.set()
sns.lineplot(K,E)
print(E)
```

```
[0.0, 0.1, 0.16, 0.15, 0.14]
```



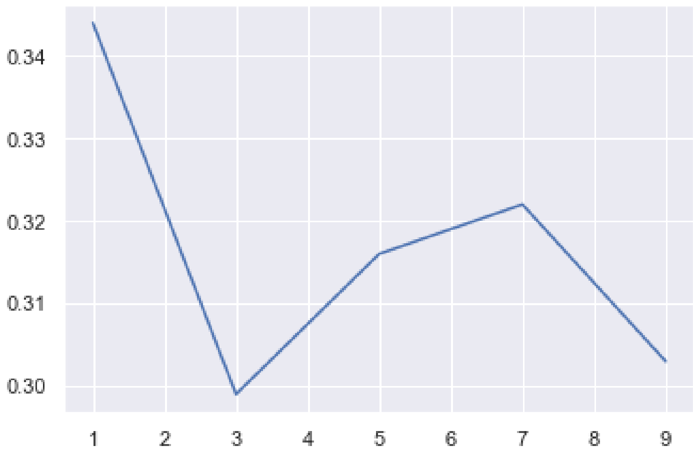
Q11: Ein increases at first then decreases.

In [120]:

```
#Q12
K = [1,3,5,7,9]
E = []
for k in K:
    E.append(err(dftrain, dftest, k, k_nbor))

sns.set()
sns.lineplot(K,E)
print(E)
```

```
[0.344, 0.299, 0.316, 0.322, 0.303]
```



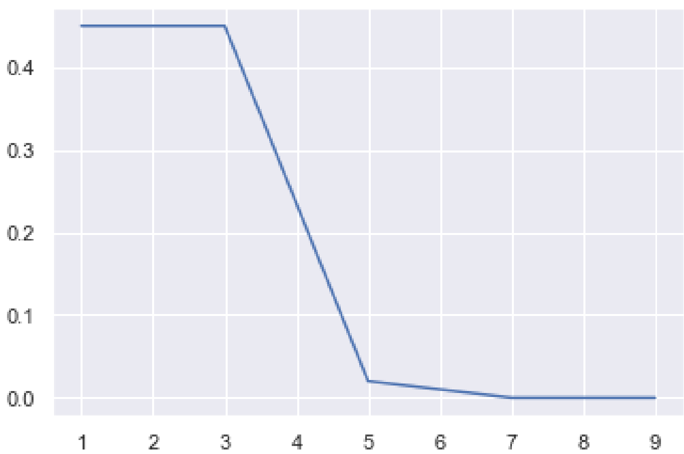
Q12: Eout does not vary much.

In [122]:

```
#Q12
Gamma = [0.001,0.1,1,10,100]
E = []
for gamma in Gamma:
    E.append(err(dftrain, dftrain, gamma, uniform))

sns.set()
sns.lineplot(K,E)
print(E)
```

```
[0.45, 0.45, 0.02, 0.0, 0.0]
```

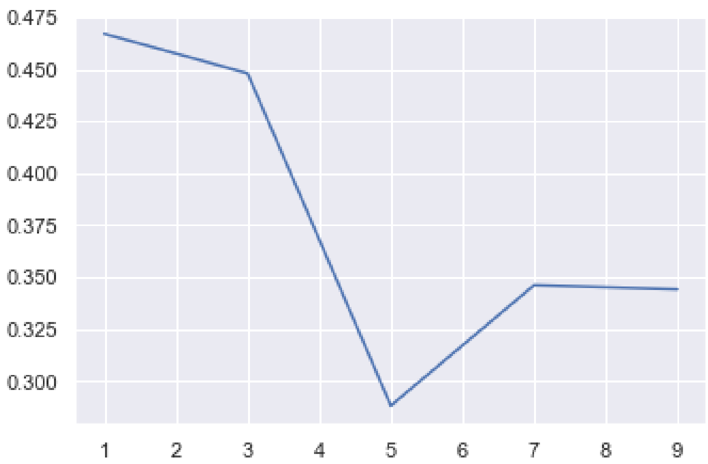


Q13: Ein decreases to 0.

In [124]:
```
#Q12
Gamma = [0.001,0.1,1,10,100]
E = []
for gamma in Gamma:
    E.append(err(dftrain, dftest, gamma, uniform))

sns.set()
sns.lineplot(K,E)
print(E)
```

```
[0.467, 0.448, 0.288, 0.346, 0.344]
```



Q14: Eout decreases to its lowest then surges.

In [ ]:

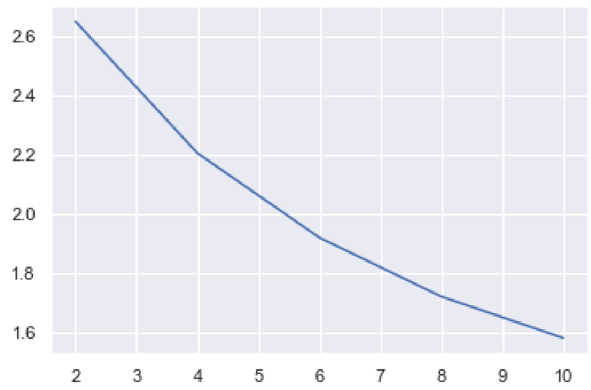
In [12]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.cluster import KMeans
```

In [11]:
```
df= np.loadtxt('hw4_nolabel_train.dat')
```

In [47]:
```
E = []
V = []
K = [2,4,6,8,10]
for k in K:
    err = 0
    sq = 0
    for T in range(500):
        kmeans = KMeans(n_clusters=k, random_state=T).fit(df)
        err += kmeans.score(df)/100/500
        sq += ((kmeans.score(df)/100)**2)/500
    E.append(-1*err)
    V.append(sq - err**2)
```

In [56]:
```
sns.set()
sns.lineplot(K,E)
print(E)
```

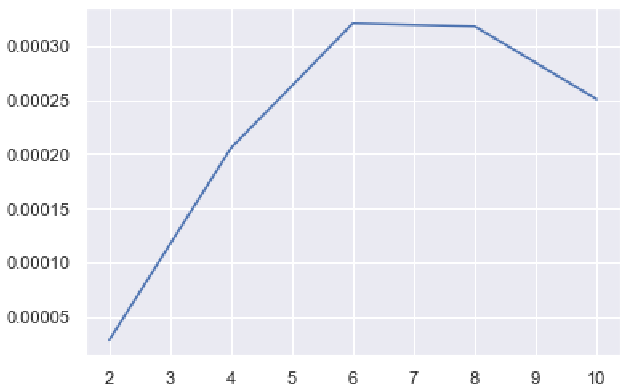
```
[2.6482794225946558, 2.2033466623852838, 1.9180549102254723, 1.7188274310568472, 1.57910757047811]
```



Q15: Average of err for 500 times decreases as k increases.

In [58]:
```
sns.set()
sns.lineplot(K,V)
print(V)
```

```
[2.8007285915343516e-05, 0.0002057968897553053, 0.00032129635692879077, 0.0003182846984906007, 0.00
025101657672488287]
```



Q16: Variance of err for 500 times increases until k = 6, then decreases.

1. The sum of weights.

$$= 10 * d^{(1)} + (d^{(1)}+1)\, d^{(2)} + \cdots + (d^{(L-2)}+1)(d^{(L-1)}) + d^{(L-1)} \times 1$$

$$\text{s.t.} \sum_{t=1}^{L-1} d^{(t)}+1 = 36.$$

for minimum weight number

if $d^{(1)} = d^{(2)} = \cdots d^{(18)} = 1$

$\therefore$ min # of weight $= \underbrace{10 + 2 + \cdots + 2 + 2}_{18\text{个}} = 46$

2. For maximum sum of weight numbers.

$\Rightarrow$ See program. greedy search.

Max = 510

3.

$$\nabla err_n(w) = \frac{\partial (x_n - ww^T x_n)^T (x_n - ww^T x_n)}{\partial w}$$

$$= \boxed{2(x_n - ww^T x_n) \cdot \frac{\partial (x_n - ww^T x_n)}{\partial w}}$$

$$= \cancel{-2(x_n - ww^T x_n)(x_n + x_n^T) w}$$

$$= -2(x_n + x_n^T) w \,(x_n - ww^T x_n)$$

$$= 2(x_n^T w)^2 w + 2(x_n^T w)(w^T w) x_n - 4 w^T x_n x_n$$

4. $\tilde{x}_n = x_n + \varepsilon_n$ $\quad E(\varepsilon_n) = 0_n$

$\varepsilon_n \sim N(0_n, I)$

$$E_{in} = \frac{1}{N}\sum_{n=1}^{N} (x_n - ww^T x_n)^T (x_n - ww^T x_n)^T$$

$$= \frac{2}{N}\sum (x_n - ww^T x_n)^T (ww^T \varepsilon_n)$$

$$+ \sum (ww^T \varepsilon_n)^T (ww^T \varepsilon_n)$$

$$\Omega(w) = E\Big[2\frac{1}{N}\sum (x_n - \cancel{w^T w^T x_n})^T (ww^T \varepsilon_n) \xrightarrow{} 0$$

$$+ \frac{1}{N}\sum (ww^T \varepsilon_n)^T (ww^T \varepsilon_n) \Big]$$

$$= \frac{1}{N}\sum E[\varepsilon_n^T ww^T ww^T \varepsilon_n]$$

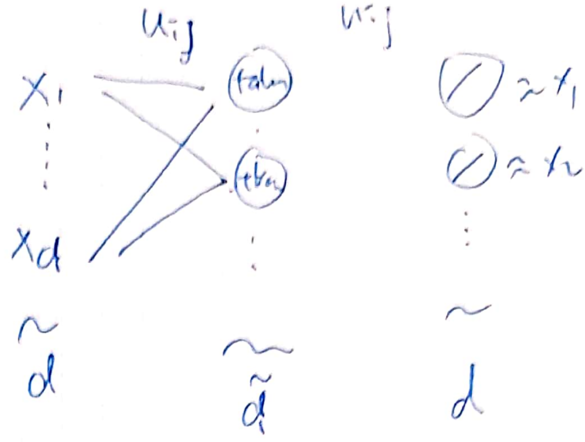
$$= w^T w \frac{1}{N}\sum E[\varepsilon_n^T ww^T \varepsilon_n)$$

$$= w^T w \frac{1}{N}\sum E[\sum_{i=1}^{d} \varepsilon_{ni}^2 w_i^2]$$

$$= w^T w \frac{1}{N}\sum^{N}\sum^{d} w_i^2 E(\varepsilon_{ni}^2)$$

$$= w^T w \frac{1}{N}\sum^{N}\sum^{d} w_i^2 = \frac{1}{N}\sum (w^T w)^2 = \underline{(w^T w)^2}.$$

5.



error function

$$= \sum_{i=1}^{d} (g_i(x) - x_i)^2$$

$$= \sum_{j=1}^{d} \Big(\sum_{k=1}^{\tilde{d}} \underbrace{u_{ki}}_{w_{ji}^{(2)}} \tanh\big(\sum_{i=1}^{d} \underbrace{u_{ij}}_{w_{ij}^{(1)}} x_i\big) - x_j \Big)^2$$

6.

$$\frac{\partial E}{\partial u_{ij}} = 2\sum^{d} \Big(\sum^{\tilde{d}} u_{ki} \tanh\big(\sum^{d} u_{ij} x_i\big) - x_j \Big)$$

$$\Big[\tanh\big(\sum^{d} u_{ij} x_i\big) + \sum_{i=1}^{\tilde{d}} u_{ki} \tanh\big(\sum u_{ij} x_i\big) \tanh'\big(\sum u_{ij} x_i\big) x_i \Big]$$

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = 2\sum^{d} \Big(\sum^{d} u_{ki} \tanh\big(\sum u_{ij} x_i\big) - x_j \Big) \cdot \Big(\sum^{\tilde{d}} u_{ki} \tanh\big(\sum u_{ij} x_i\big) \Big) x_i$$

$\tanh'(\sum u_{ij} x_i)$

$$+$$

$$\frac{\partial E}{\partial w_{ij}^{(2)}} = 2\sum^{d} \Big(\sum^{d} u_{ki} \tanh\big(\sum_i u_{ij} x_i\big) - x_j \Big) \cdot \tanh\big(\sum w_{ij}^{(1)} x_i\big)$$

$$\|$$

$$2\sum^{d} \Big(\sum^{d} u_{ki} \tanh\big(\sum u_{ij} x_i\big) - x_j \Big)$$

$$\Big[\sum^{\tilde{d}} u_{ki} \tanh\big(\sum_i u_{ij} x_i\big) \tanh'\big(\sum u_{ij} x_i\big) x_i \Big] + \tanh\big(\sum w_{ij}^{(1)} x_i\big)$$

Double A

7.

$$g_{LIN}(x) = \text{sign}\left(\|X - X_-\|^2 - \|X - X_+\|^2 \right)$$

$$= \text{sign}\left((X - X_-)^T (X - X_-) - (X - X_+)^T (X - X_+) \right)$$

$$= \text{sign}\left(2(X_+ - X_-)^T X + \|X_-\|^2 - \|X_+\|^2 \right)$$

8. ϕ if $g_{RBFNET}(x) \geq 1$

$$\Rightarrow \beta_+ \exp(-\|x - \mu_+\|^2) + \beta_- \exp(-\|x - \mu_-\|^2) \geq 0$$

$$\Rightarrow \frac{\exp(-\|x - \mu_+\|^2)}{\exp(-\|x - \mu_-\|^2)} \geq -\frac{\beta_-}{\beta_+}$$

$$\Rightarrow \underline{\|x - \mu_-\|^2 - \|x - \mu_+\|^2} \geq \ln\left(-\frac{\beta_-}{\beta_+}\right)$$

as g_{LIN}, using (7) solution

$$w = 2(\mu_+ - \mu_-)$$

$$b = \|\mu_-\|^2 - \|\mu_+\|^2 - \ln\left(-\frac{\beta_-}{\beta_+}\right)$$

9.

$$\sum_{m=1}^{M} \sum_{\substack{(x_n, r_{nm}) \\ \in D_m}} (r_{nm} - w_m^T v_n)^2$$

FOC: $$2 \sum_{\substack{(x_n, r_{nm}) \\ \in D_m}} (r_{nm} - w_m^T v_n) v_n = 0$$

$$\Rightarrow \sum (r_{nm} - w_m^T v_n) = 0$$

$$\Rightarrow w_m = \frac{\sum_n r_{nm}}{\sum_n v_n} = \underbrace{\frac{\sum_n r_{nm}}{n}}_{\text{average rating}}.$$

10.

$$\max_m v_{N+1}^T w_m = \max_m \left(\frac{1}{N} \sum v_n\right)^T w_m$$

$$= \max_m \frac{1}{N} \sum v_n^T w_m = \max \underbrace{\frac{1}{N} \sum^{N} r_{nm}}_{\text{maximum of average rating}}.$$