sml-assignment-10

October 31, 2024

[1]:
```python
import pandas as pd
df=pd.read_csv('/content/breast_cancer_survival.csv')
df
```

[1]:
```
     Age  Gender  Protein1  Protein2  Protein3  Protein4 Tumour_Stage  \
0     42  FEMALE  0.952560   2.15000  0.007972 -0.048340           II
1     54  FEMALE  0.000000   1.38020 -0.498030 -0.507320           II
2     63  FEMALE -0.523030   1.76400 -0.370190  0.010815           II
3     78  FEMALE -0.876180   0.12943 -0.370380  0.132190            I
4     42  FEMALE  0.226110   1.74910 -0.543970 -0.390210           II
..   …     …       …       …       …        …
329   59  FEMALE  0.024598   1.40050  0.024751  0.280320           II
330   41  FEMALE  0.100120  -0.46547  0.472370 -0.523870            I
331   54  FEMALE  0.753820   1.64250 -0.332850  0.857860           II
332   74  FEMALE  0.972510   1.42680 -0.366570 -0.107820           II
333   66  FEMALE  0.286380   1.39980  0.318830  0.836050           II

                          Histology ER status PR status HER2 status  \
0     Infiltrating Ductal Carcinoma  Positive  Positive    Negative
1     Infiltrating Ductal Carcinoma  Positive  Positive    Negative
2     Infiltrating Ductal Carcinoma  Positive  Positive    Negative
3     Infiltrating Ductal Carcinoma  Positive  Positive    Negative
4     Infiltrating Ductal Carcinoma  Positive  Positive    Positive
..                              …       …       …          …
329   Infiltrating Ductal Carcinoma  Positive  Positive    Positive
330   Infiltrating Ductal Carcinoma  Positive  Positive    Positive
331   Infiltrating Ductal Carcinoma  Positive  Positive    Negative
332  Infiltrating Lobular Carcinoma  Positive  Positive    Negative
333   Infiltrating Ductal Carcinoma  Positive  Positive    Negative

                   Surgery_type Date_of_Surgery Date_of_Last_Visit  \
0                         Other       20-May-18          26-Aug-18
1                         Other       26-Apr-18          25-Jan-19
2                    Lumpectomy       24-Aug-18          08-Apr-20
3                         Other       16-Nov-18          28-Jul-20
4                    Lumpectomy       12-Dec-18          05-Jan-19
..                          …             …                 …
```

```
329                   Lumpectomy       15-Jan-19          27-Mar-20
330  Modified Radical Mastectomy       25-Jul-18          23-Apr-19
331            Simple Mastectomy       26-Mar-19          11-Oct-19
332                   Lumpectomy       26-Nov-18          05-Dec-18
333  Modified Radical Mastectomy       04-Feb-19          10-Aug-19

    Patient_Status
0            Alive
1             Dead
2            Alive
3            Alive
4            Alive
..             …
329          Alive
330          Alive
331           Dead
332          Alive
333           Dead

[334 rows x 15 columns]
```

[2]:

```
df['Gender']=df['Gender'].replace(['FEMALE','MALE'],[1,0])
df['Tumour_Stage']=df['Tumour_Stage'].replace(['I','II','III'],[0,1,2])
df['ER status']=df['ER status'].replace(['Positive','Negative'],[1,0])
df['PR status']=df['PR status'].replace(['Positive','Negative'],[1,0])
df['HER2 status']=df['HER2 status'].replace(['Positive','Negative'],[1,0])
df['Patient_Status']=df['Patient_Status'].replace(['Alive','Dead'],[1,0])
df['Surgery_type']=df['Surgery_type'].replace(['Lumpectomy','Modified Radical␣
↪Mastectomy','Simple Mastectomy','Other'],[0,1,2,3])
df['Histology']=df['Histology'].replace(['Infiltrating Ductal␣
↪Carcinoma','Infiltrating Lobular Carcinoma','Mucinous Carcinoma'],[0,1,2])
df
```

```
<ipython-input-2-d66e4b74289f>:1: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['Gender']=df['Gender'].replace(['FEMALE','MALE'],[1,0])
<ipython-input-2-d66e4b74289f>:2: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['Tumour_Stage']=df['Tumour_Stage'].replace(['I','II','III'],[0,1,2])
<ipython-input-2-d66e4b74289f>:3: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
```

```
  df['ER status']=df['ER status'].replace(['Positive','Negative'],[1,0])
<ipython-input-2-d66e4b74289f>:4: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['PR status']=df['PR status'].replace(['Positive','Negative'],[1,0])
<ipython-input-2-d66e4b74289f>:5: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['HER2 status']=df['HER2 status'].replace(['Positive','Negative'],[1,0])
<ipython-input-2-d66e4b74289f>:6: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['Patient_Status']=df['Patient_Status'].replace(['Alive','Dead'],[1,0])
<ipython-input-2-d66e4b74289f>:7: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['Surgery_type']=df['Surgery_type'].replace(['Lumpectomy','Modified Radical
Mastectomy','Simple Mastectomy','Other'],[0,1,2,3])
<ipython-input-2-d66e4b74289f>:8: FutureWarning: Downcasting behavior in
`replace` is deprecated and will be removed in a future version. To retain the
old behavior, explicitly call `result.infer_objects(copy=False)`. To opt-in to
the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df['Histology']=df['Histology'].replace(['Infiltrating Ductal
Carcinoma','Infiltrating Lobular Carcinoma','Mucinous Carcinoma'],[0,1,2])
```

[2]:

```
     Age  Gender  Protein1  Protein2  Protein3  Protein4  Tumour_Stage  \
0     42       1  0.952560   2.15000  0.007972 -0.048340             1
1     54       1  0.000000   1.38020 -0.498030 -0.507320             1
2     63       1 -0.523030   1.76400 -0.370190  0.010815             1
3     78       1 -0.876180   0.12943 -0.370380  0.132190             0
4     42       1  0.226110   1.74910 -0.543970 -0.390210             1
..   …     …       …        …        …        …         …
329   59       1  0.024598   1.40050  0.024751  0.280320             1
330   41       1  0.100120  -0.46547  0.472370 -0.523870             0
331   54       1  0.753820   1.64250 -0.332850  0.857860             1
332   74       1  0.972510   1.42680 -0.366570 -0.107820             1
333   66       1  0.286380   1.39980  0.318830  0.836050             1

     Histology  ER status  PR status  HER2 status  Surgery_type  \
0            0          1          1            0             3
1            0          1          1            0             3
2            0          1          1            0             0
3            0          1          1            0             3
```

```
4           0          1          1           1            0
..          …          …          …           …            …
329         0          1          1           1            0
330         0          1          1           1            1
331         0          1          1           0            2
332         1          1          1           0            0
333         0          1          1           0            1

    Date_of_Surgery Date_of_Last_Visit  Patient_Status
0         20-May-18          26-Aug-18             1.0
1         26-Apr-18          25-Jan-19             0.0
2         24-Aug-18          08-Apr-20             1.0
3         16-Nov-18          28-Jul-20             1.0
4         12-Dec-18          05-Jan-19             1.0
..              …                  …             …
329       15-Jan-19          27-Mar-20             1.0
330       25-Jul-18          23-Apr-19             1.0
331       26-Mar-19          11-Oct-19             0.0
332       26-Nov-18          05-Dec-18             1.0
333       04-Feb-19          10-Aug-19             0.0

[334 rows x 15 columns]
```

[3]:
```python
df.fillna(0, inplace=True)
df
```

[3]:
```
     Age  Gender  Protein1  Protein2  Protein3  Protein4  Tumour_Stage  \
0     42       1  0.952560   2.15000  0.007972 -0.048340             1
1     54       1  0.000000   1.38020 -0.498030 -0.507320             1
2     63       1 -0.523030   1.76400 -0.370190  0.010815             1
3     78       1 -0.876180   0.12943 -0.370380  0.132190             0
4     42       1  0.226110   1.74910 -0.543970 -0.390210             1
..   …      …        …        …        …        …            …
329   59       1  0.024598   1.40050  0.024751  0.280320             1
330   41       1  0.100120  -0.46547  0.472370 -0.523870             0
331   54       1  0.753820   1.64250 -0.332850  0.857860             1
332   74       1  0.972510   1.42680 -0.366570 -0.107820             1
333   66       1  0.286380   1.39980  0.318830  0.836050             1

     Histology  ER status  PR status  HER2 status  Surgery_type  \
0            0          1          1            0             3
1            0          1          1            0             3
2            0          1          1            0             0
3            0          1          1            0             3
4            0          1          1            1             0
..          …          …          …           …            …
329          0          1          1            1             0
```

```
330          0          1          1          1            1
331          0          1          1          0            2
332          1          1          1          0            0
333          0          1          1          0            1

    Date_of_Surgery Date_of_Last_Visit  Patient_Status
0         20-May-18          26-Aug-18             1.0
1         26-Apr-18          25-Jan-19             0.0
2         24-Aug-18          08-Apr-20             1.0
3         16-Nov-18          28-Jul-20             1.0
4         12-Dec-18          05-Jan-19             1.0
..              …                  …             …
329       15-Jan-19          27-Mar-20             1.0
330       25-Jul-18          23-Apr-19             1.0
331       26-Mar-19          11-Oct-19             0.0
332       26-Nov-18          05-Dec-18             1.0
333       04-Feb-19          10-Aug-19             0.0

[334 rows x 15 columns]
```

[4]:
```python
y=df['Patient_Status']
y
```

[4]:
```
0      1.0
1      0.0
2      1.0
3      1.0
4      1.0
      …
329    1.0
330    1.0
331    0.0
332    1.0
333    0.0
Name: Patient_Status, Length: 334, dtype: float64
```

[5]:
```python
x=df.drop(['Patient_Status','Date_of_Last_Visit','Date_of_Surgery'],axis=1)
x
```

[5]:
```
     Age  Gender  Protein1  Protein2  Protein3  Protein4  Tumour_Stage  \
0     42       1  0.952560   2.15000  0.007972 -0.048340             1
1     54       1  0.000000   1.38020 -0.498030 -0.507320             1
2     63       1 -0.523030   1.76400 -0.370190  0.010815             1
3     78       1 -0.876180   0.12943 -0.370380  0.132190             0
4     42       1  0.226110   1.74910 -0.543970 -0.390210             1
..   …     …       …       …       …       …           …
329   59       1  0.024598   1.40050  0.024751  0.280320             1
```

```
330   41        1  0.100120  -0.46547   0.472370 -0.523870             0
331   54        1  0.753820   1.64250  -0.332850  0.857860             1
332   74        1  0.972510   1.42680  -0.366570 -0.107820             1
333   66        1  0.286380   1.39980   0.318830  0.836050             1

     Histology  ER status  PR status  HER2 status  Surgery_type
0            0          1          1            0             3
1            0          1          1            0             3
2            0          1          1            0             0
3            0          1          1            0             3
4            0          1          1            1             0
..         …          …          …            …             …
329          0          1          1            1             0
330          0          1          1            1             1
331          0          1          1            0             2
332          1          1          1            0             0
333          0          1          1            0             1

[334 rows x 12 columns]
```

[41]:

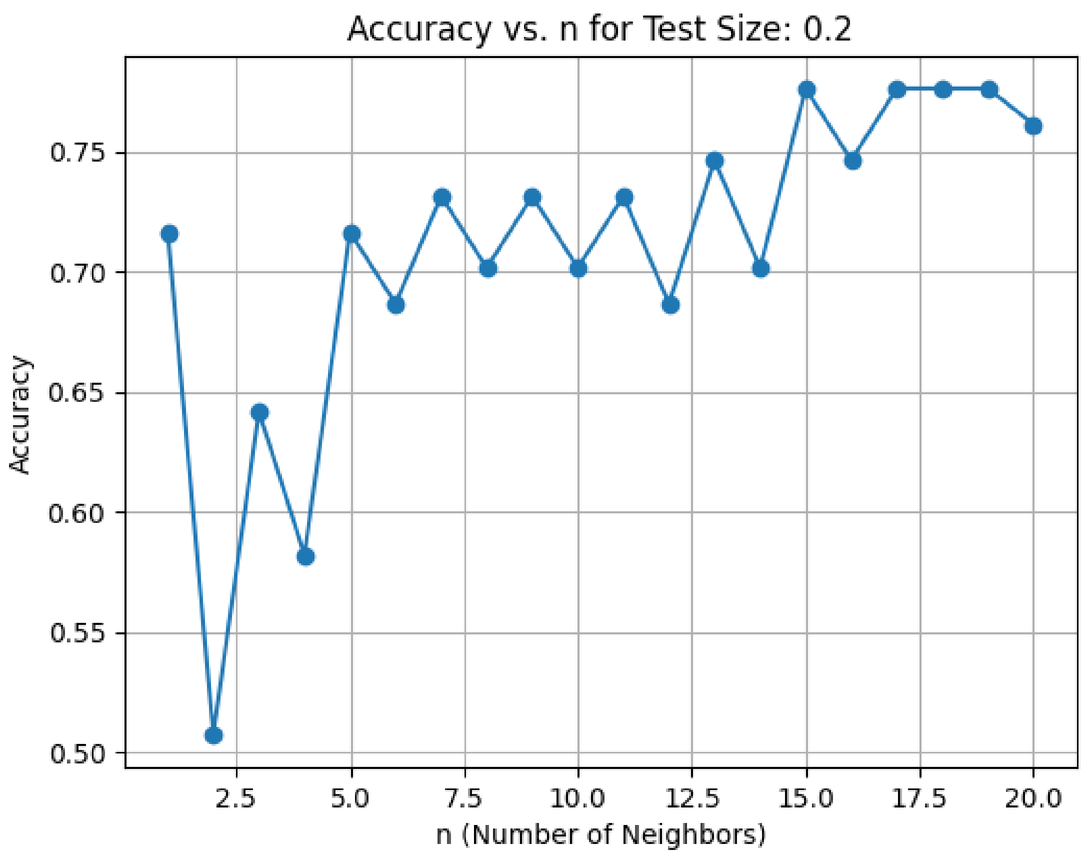
```python
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
def diff_train_knn(x, y, test_sizes, random_state):
    results = {}
    for test_size in test_sizes:
        print(f"\nTest Size: {test_size}")
        results[test_size] = {}
        x_train, x_test, y_train, y_test = train_test_split(x, y,␣
↪test_size=test_size, random_state=random_state)
        scaler = StandardScaler()
        x_train = scaler.fit_transform(x_train)
        x_test = scaler.transform(x_test)
        for n in range(1, 21):
            model = KNeighborsClassifier(n_neighbors=n)
            model.fit(x_train, y_train)
            y_pred = model.predict(x_test)
            accuracy = accuracy_score(y_test, y_pred)
            print(f"n = {n}, Accuracy = {accuracy:.4f}")
            results[test_size][n] = accuracy
    return results

test_sizes = [0.2,0.25, 0.3, 0.35]

results = diff_train_knn(x, y, test_sizes, random_state=42)
```

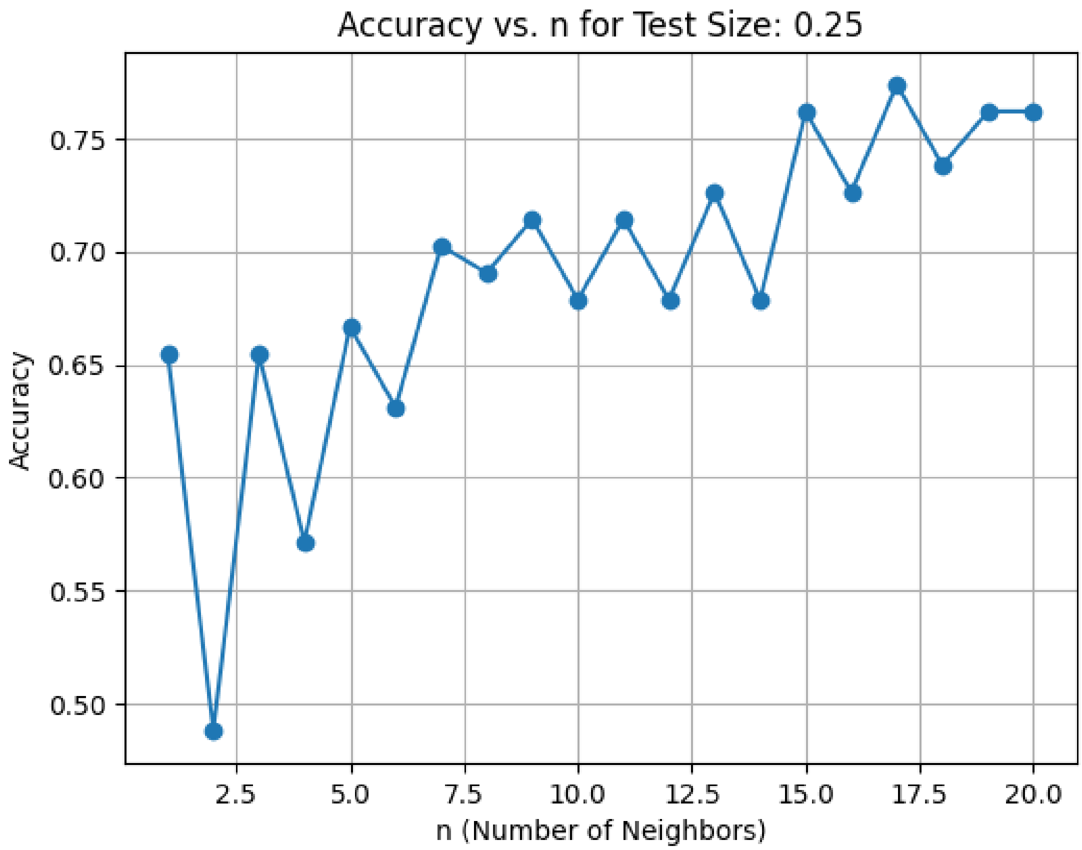
```
Test Size: 0.2
n = 1, Accuracy = 0.7164
n = 2, Accuracy = 0.5075
n = 3, Accuracy = 0.6418
n = 4, Accuracy = 0.5821
n = 5, Accuracy = 0.7164
n = 6, Accuracy = 0.6866
n = 7, Accuracy = 0.7313
n = 8, Accuracy = 0.7015
n = 9, Accuracy = 0.7313
n = 10, Accuracy = 0.7015
n = 11, Accuracy = 0.7313
n = 12, Accuracy = 0.6866
n = 13, Accuracy = 0.7463
n = 14, Accuracy = 0.7015
n = 15, Accuracy = 0.7761
n = 16, Accuracy = 0.7463
n = 17, Accuracy = 0.7761
n = 18, Accuracy = 0.7761
n = 19, Accuracy = 0.7761
n = 20, Accuracy = 0.7612

Test Size: 0.25
n = 1, Accuracy = 0.6548
n = 2, Accuracy = 0.4881
n = 3, Accuracy = 0.6548
n = 4, Accuracy = 0.5714
n = 5, Accuracy = 0.6667
n = 6, Accuracy = 0.6310
n = 7, Accuracy = 0.7024
n = 8, Accuracy = 0.6905
n = 9, Accuracy = 0.7143
n = 10, Accuracy = 0.6786
n = 11, Accuracy = 0.7143
n = 12, Accuracy = 0.6786
n = 13, Accuracy = 0.7262
n = 14, Accuracy = 0.6786
n = 15, Accuracy = 0.7619
n = 16, Accuracy = 0.7262
n = 17, Accuracy = 0.7738
n = 18, Accuracy = 0.7381
n = 19, Accuracy = 0.7619
n = 20, Accuracy = 0.7619

Test Size: 0.3
n = 1, Accuracy = 0.6535
n = 2, Accuracy = 0.5248
```

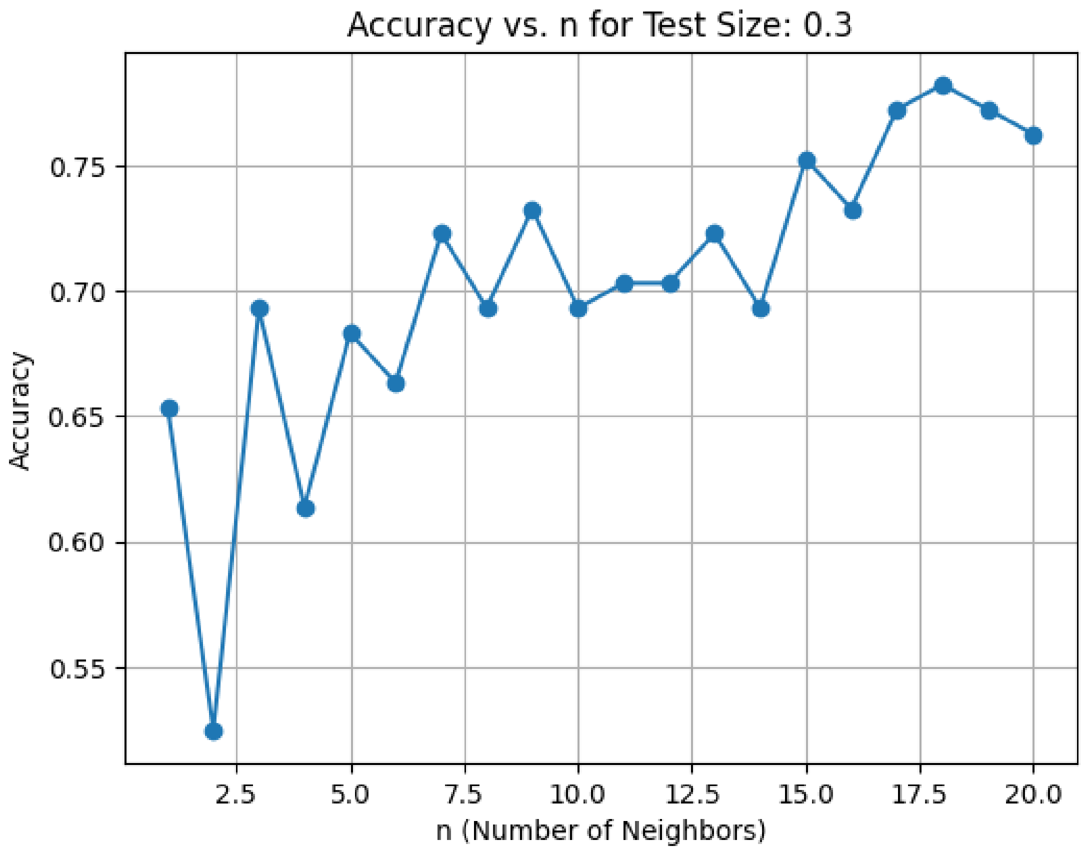
```
n = 3, Accuracy = 0.6931
n = 4, Accuracy = 0.6139
n = 5, Accuracy = 0.6832
n = 6, Accuracy = 0.6634
n = 7, Accuracy = 0.7228
n = 8, Accuracy = 0.6931
n = 9, Accuracy = 0.7327
n = 10, Accuracy = 0.6931
n = 11, Accuracy = 0.7030
n = 12, Accuracy = 0.7030
n = 13, Accuracy = 0.7228
n = 14, Accuracy = 0.6931
n = 15, Accuracy = 0.7525
n = 16, Accuracy = 0.7327
n = 17, Accuracy = 0.7723
n = 18, Accuracy = 0.7822
n = 19, Accuracy = 0.7723
n = 20, Accuracy = 0.7624

Test Size: 0.35
n = 1, Accuracy = 0.6496
n = 2, Accuracy = 0.5128
n = 3, Accuracy = 0.6752
n = 4, Accuracy = 0.6239
n = 5, Accuracy = 0.6923
n = 6, Accuracy = 0.6410
n = 7, Accuracy = 0.7094
n = 8, Accuracy = 0.6838
n = 9, Accuracy = 0.7179
n = 10, Accuracy = 0.6752
n = 11, Accuracy = 0.7009
n = 12, Accuracy = 0.6923
n = 13, Accuracy = 0.7094
n = 14, Accuracy = 0.6923
n = 15, Accuracy = 0.7350
n = 16, Accuracy = 0.7350
n = 17, Accuracy = 0.7692
n = 18, Accuracy = 0.7692
n = 19, Accuracy = 0.7778
n = 20, Accuracy = 0.7778
```

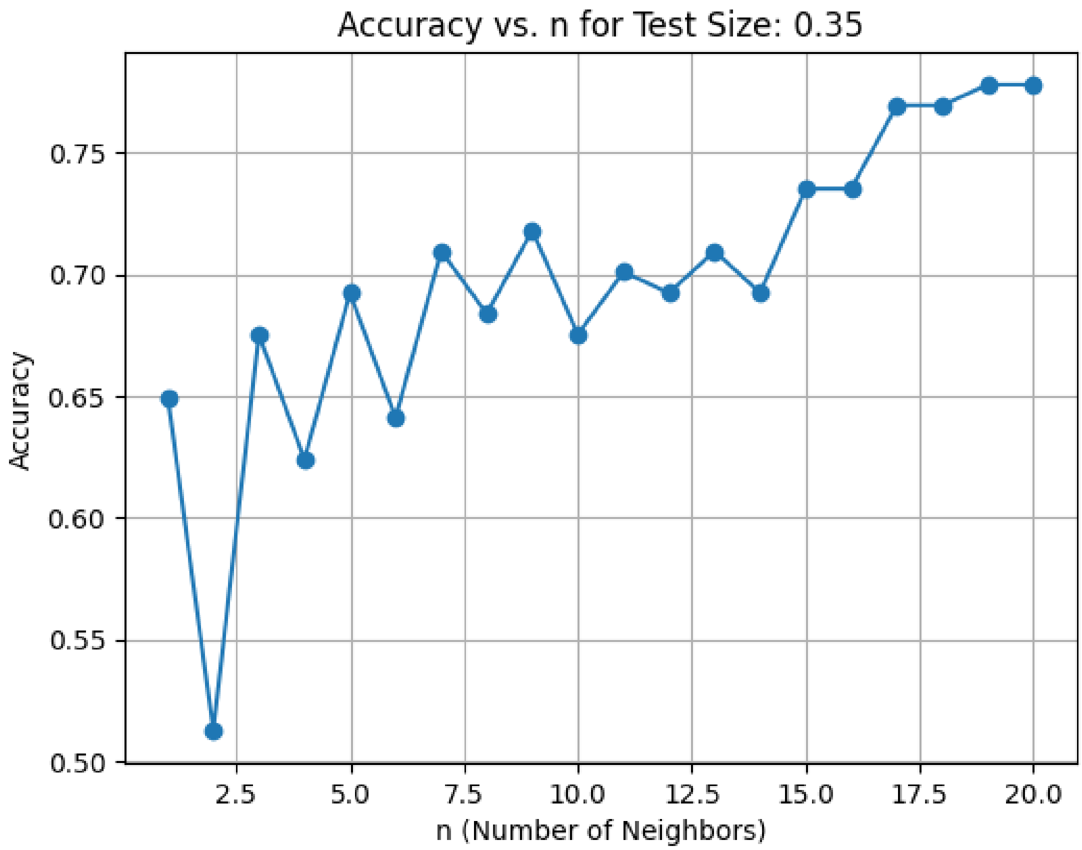
[42]:
```
import matplotlib.pyplot as plt
for test_size, accuracy_dict in results.items():
  n_values = list(accuracy_dict.keys())
  accuracy_values = list(accuracy_dict.values())
  plt.figure()
  plt.plot(n_values, accuracy_values, marker='o')
```

```python
plt.title(f"Accuracy vs. n for Test Size: {test_size}")
plt.xlabel("n (Number of Neighbors)")
plt.ylabel("Accuracy")
plt.grid(True)
plt.show()
```




Accuracy vs. n for Test Size: 0.25
Accuracy
0.75
0.70
0.65
0.60
0.55
0.50
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
n (Number of Neighbors)


Accuracy vs. n for Test Size: 0.3
Accuracy
0.55
0.60
0.65
0.70
0.75
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
n (Number of Neighbors)



[37]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)
from sklearn.svm import SVC
svc=SVC()
svc.fit(x_train,y_train)
y_pred=svc.predict(x_test)
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[37]: 0.7313432835820896

[38]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
 ↪25,random_state=0)
from sklearn.svm import SVC
svc=SVC()
svc.fit(x_train,y_train)
y_pred=svc.predict(x_test)
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[38]: 0.7261904761904762

[39]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3,random_state=0)
from sklearn.svm import SVC
svc=SVC()
svc.fit(x_train,y_train)
y_pred=svc.predict(x_test)
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[39]: 0.7326732673267327

[40]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
 ↪35,random_state=0)
from sklearn.svm import SVC
svc=SVC()
svc.fit(x_train,y_train)
y_pred=svc.predict(x_test)
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

[40]: 0.7606837606837606

[ ]:
```

```