In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
import re
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
```

Stations

In [2]:

```
a=pd.read_csv(r"C:\Users\user\Downloads\C10_air\stations.csv")
a
```

Out[2]:

	id	name	address	lon	lat	elevation
0	28079004	Pza. de España	Plaza de España	-3.712247	40.423853	635
1	28079008	Escuelas Aguirre	Entre C/ Alcalá y C/ O' Donell	-3.682319	40.421564	670
2	28079011	Avda. Ramón y Cajal	Avda. Ramón y Cajal esq. C/ Príncipe de Vergara	-3.677356	40.451475	708
3	28079016	Arturo Soria	C/ Arturo Soria esq. C/ Vizconde de los Asilos	-3.639233	40.440047	693
4	28079017	Villaverde	C/. Juan Peñalver	-3.713322	40.347139	604
5	28079018	Farolillo	Calle Farolillo - C/Ervigio	-3.731853	40.394781	630
6	28079024	Casa de Campo	Casa de Campo (Terminal del Teleférico)	-3.747347	40.419356	642
7	28079027	Barajas Pueblo	C/. Júpiter, 21 (Barajas)	-3.580031	40.476928	621
8	28079035	Pza. del Carmen	Plaza del Carmen esq. Tres Cruces.	-3.703172	40.419208	659
9	28079036	Moratalaz	Avd. Moratalaz esq. Camino de los Vinateros	-3.645306	40.407947	685
10	28079038	Cuatro Caminos	Avda. Pablo Iglesias esq. C/ Marqués de Lema	-3.707128	40.445544	698
11	28079039	Barrio del Pilar	Avd. Betanzos esq. C/ Monforte de Lemos	-3.711542	40.478228	674
12	28079040	Vallecas	C/ Arroyo del Olivar esq. C/ Río Grande.	-3.651522	40.388153	677
13	28079047	Mendez Alvaro	C/ Juan de Mariana / Pza. Amanecer Mendez Alvaro	-3.686825	40.398114	599
14	28079048	Castellana	C/ Jose Gutierrez Abascal	-3.690367	40.439897	676
15	28079049	Parque del Retiro	Paseo Venezuela- Casa de Vacas	-3.682583	40.414444	662
16	28079050	Plaza Castilla	Plaza Castilla (Canal)	-3.688769	40.465572	728
17	28079054	Ensanche de Vallecas	Avda La Gavia / Avda. Las Suertes	-3.612117	40.372933	627
18	28079055	Urb. Embajada	C/ Riaño (Barajas)	-3.580747	40.462531	618
19	28079056	Pza. Fernández Ladreda	Pza. Fernández Ladreda - Avda. Oporto	-3.718728	40.384964	604
20	28079057	Sanchinarro	C/ Princesa de Eboli esq C/ Maria Tudor	-3.660503	40.494208	700
21	28079058	El Pardo	Avda. La Guardia	-3.774611	40.518058	615
22	28079059	Juan Carlos I	Parque Juan Carlos I (frente oficinas mantenim...	-3.609072	40.465250	660
23	28079060	Tres Olivos	Plaza Tres Olivos	-3.689761	40.500589	715

In [3]:

```
a.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 6 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   id         24 non-null     int64  
 1   name       24 non-null     object 
 2   address    24 non-null     object 
 3   lon        24 non-null     float64
 4   lat        24 non-null     float64
 5   elevation  24 non-null     int64  
dtypes: float64(2), int64(2), object(2)
memory usage: 1.2+ KB
```

In [4]:

```
b=a.fillna(value=87)
b
```

Out[4]:

	id	name	address	lon	lat	elevation
0	28079004	Pza. de España	Plaza de España	-3.712247	40.423853	635
1	28079008	Escuelas Aguirre	Entre C/ Alcalá y C/ O' Donell	-3.682319	40.421564	670
2	28079011	Avda. Ramón y Cajal	Avda. Ramón y Cajal esq. C/ Príncipe de Vergara	-3.677356	40.451475	708
3	28079016	Arturo Soria	C/ Arturo Soria esq. C/ Vizconde de los Asilos	-3.639233	40.440047	693
4	28079017	Villaverde	C/. Juan Peñalver	-3.713322	40.347139	604
5	28079018	Farolillo	Calle Farolillo - C/Ervigio	-3.731853	40.394781	630
6	28079024	Casa de Campo	Casa de Campo (Terminal del Teleférico)	-3.747347	40.419356	642
7	28079027	Barajas Pueblo	C/. Júpiter, 21 (Barajas)	-3.580031	40.476928	621
8	28079035	Pza. del Carmen	Plaza del Carmen esq. Tres Cruces.	-3.703172	40.419208	659
9	28079036	Moratalaz	Avd. Moratalaz esq. Camino de los Vinateros	-3.645306	40.407947	685
10	28079038	Cuatro Caminos	Avda. Pablo Iglesias esq. C/ Marqués de Lema	-3.707128	40.445544	698
11	28079039	Barrio del Pilar	Avd. Betanzos esq. C/ Monforte de Lemos	-3.711542	40.478228	674
12	28079040	Vallecas	C/ Arroyo del Olivar esq. C/ Río Grande.	-3.651522	40.388153	677
13	28079047	Mendez Alvaro	C/ Juan de Mariana / Pza. Amanecer Mendez Alvaro	-3.686825	40.398114	599
14	28079048	Castellana	C/ Jose Gutierrez Abascal	-3.690367	40.439897	676
15	28079049	Parque del Retiro	Paseo Venezuela- Casa de Vacas	-3.682583	40.414444	662
16	28079050	Plaza Castilla	Plaza Castilla (Canal)	-3.688769	40.465572	728
17	28079054	Ensanche de Vallecas	Avda La Gavia / Avda. Las Suertes	-3.612117	40.372933	627
18	28079055	Urb. Embajada	C/ Riaño (Barajas)	-3.580747	40.462531	618
19	28079056	Pza. Fernández Ladreda	Pza. Fernández Ladreda - Avda. Oporto	-3.718728	40.384964	604
20	28079057	Sanchinarro	C/ Princesa de Eboli esq C/ Maria Tudor	-3.660503	40.494208	700
21	28079058	El Pardo	Avda. La Guardia	-3.774611	40.518058	615
22	28079059	Juan Carlos I	Parque Juan Carlos I (frente oficinas mantenim...	-3.609072	40.465250	660
23	28079060	Tres Olivos	Plaza Tres Olivos	-3.689761	40.500589	715

In [5]:

```
b.columns
```

Out[5]:

```
Index(['id', 'name', 'address', 'lon', 'lat', 'elevation'], dtype='objec
t')
```

In [6]:

```
c=b.head(10)
c
```

Out[6]:

	id	name	address	lon	lat	elevation
0	28079004	Pza. de España	Plaza de España	-3.712247	40.423853	635
1	28079008	Escuelas Aguirre	Entre C/ Alcalá y C/ O' Donell	-3.682319	40.421564	670
2	28079011	Avda. Ramón y Cajal	Avda. Ramón y Cajal esq. C/ Príncipe de Vergara	-3.677356	40.451475	708
3	28079016	Arturo Soria	C/ Arturo Soria esq. C/ Vizconde de los Asilos	-3.639233	40.440047	693
4	28079017	Villaverde	C/. Juan Peñalver	-3.713322	40.347139	604
5	28079018	Farolillo	Calle Farolillo - C/Ervigio	-3.731853	40.394781	630
6	28079024	Casa de Campo	Casa de Campo (Terminal del Teleférico)	-3.747347	40.419356	642
7	28079027	Barajas Pueblo	C/. Júpiter, 21 (Barajas)	-3.580031	40.476928	621
8	28079035	Pza. del Carmen	Plaza del Carmen esq. Tres Cruces.	-3.703172	40.419208	659
9	28079036	Moratalaz	Avd. Moratalaz esq. Camino de los Vinateros	-3.645306	40.407947	685

In [7]:

```
d=c[['id', 'lon', 'lat', 'elevation']]
d
```

Out[7]:

	id	lon	lat	elevation
0	28079004	-3.712247	40.423853	635
1	28079008	-3.682319	40.421564	670
2	28079011	-3.677356	40.451475	708
3	28079016	-3.639233	40.440047	693
4	28079017	-3.713322	40.347139	604
5	28079018	-3.731853	40.394781	630
6	28079024	-3.747347	40.419356	642
7	28079027	-3.580031	40.476928	621
8	28079035	-3.703172	40.419208	659
9	28079036	-3.645306	40.407947	685

In [101]:

```
x=b[['id', 'lon', 'lat']]
y=b['elevation']
```

In [120]:

```
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=4)
```

In [121]:

```
from sklearn.linear_model import LinearRegression

lr=LinearRegression()
lr.fit(x_train,y_train)
```

Out[121]:

```
LinearRegression()
```

In [122]:

```
print(lr.score(x_test,y_test))
```

```
-4.468634918240467
```

In [123]:

```
from sklearn.linear_model import Ridge,Lasso
```

In [124]:

```
rr=Ridge(alpha=10)
rr.fit(x_train,y_train)
```

Out[124]:

```
Ridge(alpha=10)
```

In [125]:

```
rr.score(x_test,y_test)
```

Out[125]:

```
-0.11909301693932184
```

In [126]:

```
la=Lasso(alpha=10)
la.fit(x_train,y_train)
```

Out[126]:

```
Lasso(alpha=10)
```

In [127]:

```
la.score(x_test,y_test)
```

Out[127]:

```
-0.09047274405261918
```

In [128]:

```
f=StandardScaler().fit_transform(x)
```

In [129]:

```
logr=LogisticRegression()
logr.fit(f,y)
```

Out[129]:

```
LogisticRegression()
```

In [130]:

```
i=[[10,20,30]]
```

In [131]:

```
logr.predict_proba(i)[0][0]
```

Out[131]:

```
6.927458571349536e-21
```

In [132]:

```
logr.score(x_test,y_test)
```

Out[132]:

```
0.2
```

In [133]:

```
from sklearn.linear_model import ElasticNet
en=ElasticNet()
en.fit(x_train,y_train)
```

Out[133]:

```
ElasticNet()
```

In [134]:

```
prediction=en.predict(x_test)
print(en.score(x_test,y_test))
```

```
-0.11432109827336023
```

In [135]:

```
from sklearn.ensemble import RandomForestClassifier

rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[135]:

```
RandomForestClassifier()
```

In [136]:

```
parameters={'max_depth':[1,2,3,4,5],
           'min_samples_leaf':[5,10,12,34,12],
           'n_estimators':[10,20,23,56,13]
           }
```

In [137]:

```
from sklearn.model_selection import GridSearchCV

grid_search=GridSearchCV(estimator=rfc,param_grid=parameters,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

```
C:\ProgramData\Anaconda3\lib\site-packages\sklearn\model_selection\_split.
py:666: UserWarning: The least populated class in y has only 1 members, wh
ich is less than n_splits=2.
  warnings.warn(("The least populated class in y has only %d"
```

Out[137]:

```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [1, 2, 3, 4, 5],
                         'min_samples_leaf': [5, 10, 12, 34, 12],
                         'n_estimators': [10, 20, 23, 56, 13]},
             scoring='accuracy')
```

In [138]:

```
grid_search.best_score_
```

Out[138]:

```
0.05555555555555555
```

In [139]:

```
rfc_best=grid_search.best_estimator_
```

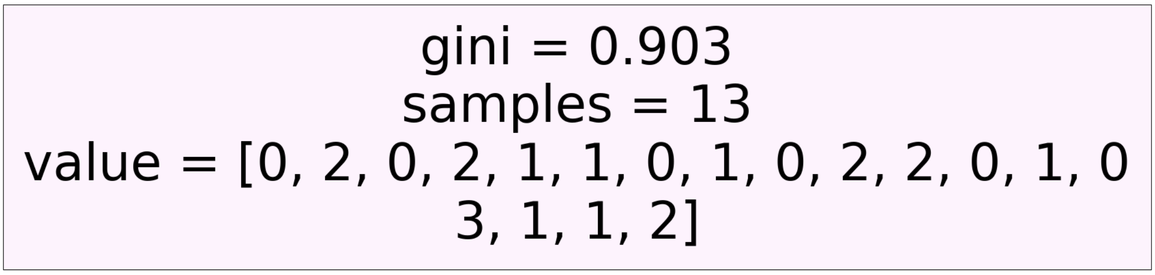
In [141]:

```
from sklearn.tree import plot_tree

plt.figure(figsize=(30,10))
plot_tree(rfc_best.estimators_[5],filled=True)
```

Out[141]:

```
[Text(837.0, 271.8, 'gini = 0.903\nsamples = 13\nvalue = [0, 2, 0, 2, 1,
1, 0, 1, 0, 2, 2, 0, 1, 0\n3, 1, 1, 2]')]
```




```
Conclusion: Logistic score=0.2.It has the highest accuracy.
```