In [1]:
```
import numpy as np
import pandas as pd
from matplotlib.pyplot import subplots
from statsmodels.datasets import get_rdataset
import sklearn.model_selection as skm
from ISLP import load_data, confusion_table
from ISLP.models import ModelSpec as MS
```

In [2]:
```
from sklearn.tree import (DecisionTreeClassifier as DTC,
DecisionTreeRegressor as DTR,
plot_tree,
export_text)
from sklearn.metrics import (accuracy_score,
log_loss)
from sklearn.ensemble import \
(RandomForestRegressor as RF,
GradientBoostingRegressor as GBR)
from ISLP.bart import BART
```

In [33]:

```
Credit = load_data('Credit')
credit_data = Credit[['ID','Income','Limit','Rating','Cards','Age','Education','Balance']]
High = np.where(Credit.Income > 100,
"Yes",
"No")
High
credit_data
```

Out[33]:

	ID	Income	Limit	Rating	Cards	Age	Education	Balance
0	1	14.891	3606	283	2	34	11	333
1	2	106.025	6645	483	3	82	15	903
2	3	104.593	7075	514	4	71	11	580
3	4	148.924	9504	681	3	36	11	964
4	5	55.882	4897	357	2	68	16	331
...	...	...	...	...	...	...	...	...
395	396	12.096	4100	307	3	32	13	560
396	397	13.364	3838	296	5	65	17	480
397	398	57.872	4171	321	5	67	12	138
398	399	37.728	2525	192	1	44	13	0
399	400	18.701	5524	415	5	64	7	966

400 rows × 8 columns

In [28]:
```
Credit
```

Out[28]:

	ID	Income	Limit	Rating	Cards	Age	Education	Gender	Student	Married	Ethnicity	Balance
0	1	14.891	3606	283	2	34	11	Male	No	Yes	Caucasian	333
1	2	106.025	6645	483	3	82	15	Female	Yes	Yes	Asian	903
2	3	104.593	7075	514	4	71	11	Male	No	No	Asian	580
3	4	148.924	9504	681	3	36	11	Female	No	No	Asian	964
4	5	55.882	4897	357	2	68	16	Male	No	Yes	Caucasian	331
...	...	...	...	...	...	...	...	...	...	...	...	...
395	396	12.096	4100	307	3	32	13	Male	No	Yes	Caucasian	560
396	397	13.364	3838	296	5	65	17	Male	No	No	African American	480
397	398	57.872	4171	321	5	67	12	Female	No	Yes	Caucasian	138
398	399	37.728	2525	192	1	44	13	Male	No	Yes	Caucasian	0
399	400	18.701	5524	415	5	64	7	Female	No	No	Asian	966

400 rows × 12 columns

In [34]:
```
model = MS(credit_data.columns.drop('Income'), intercept=False)
D = model.fit_transform(credit_data)
feature_names = list(D.columns)
X = np.asarray(D)
```

In [35]:
```
clf = DTC(criterion='entropy',
max_depth=3,
random_state=0)
clf.fit(X, High)
```

Out[35]:
```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy', max_depth=3, random_state=0)
```

In [36]:
```
accuracy_score(High, clf.predict(X))
```

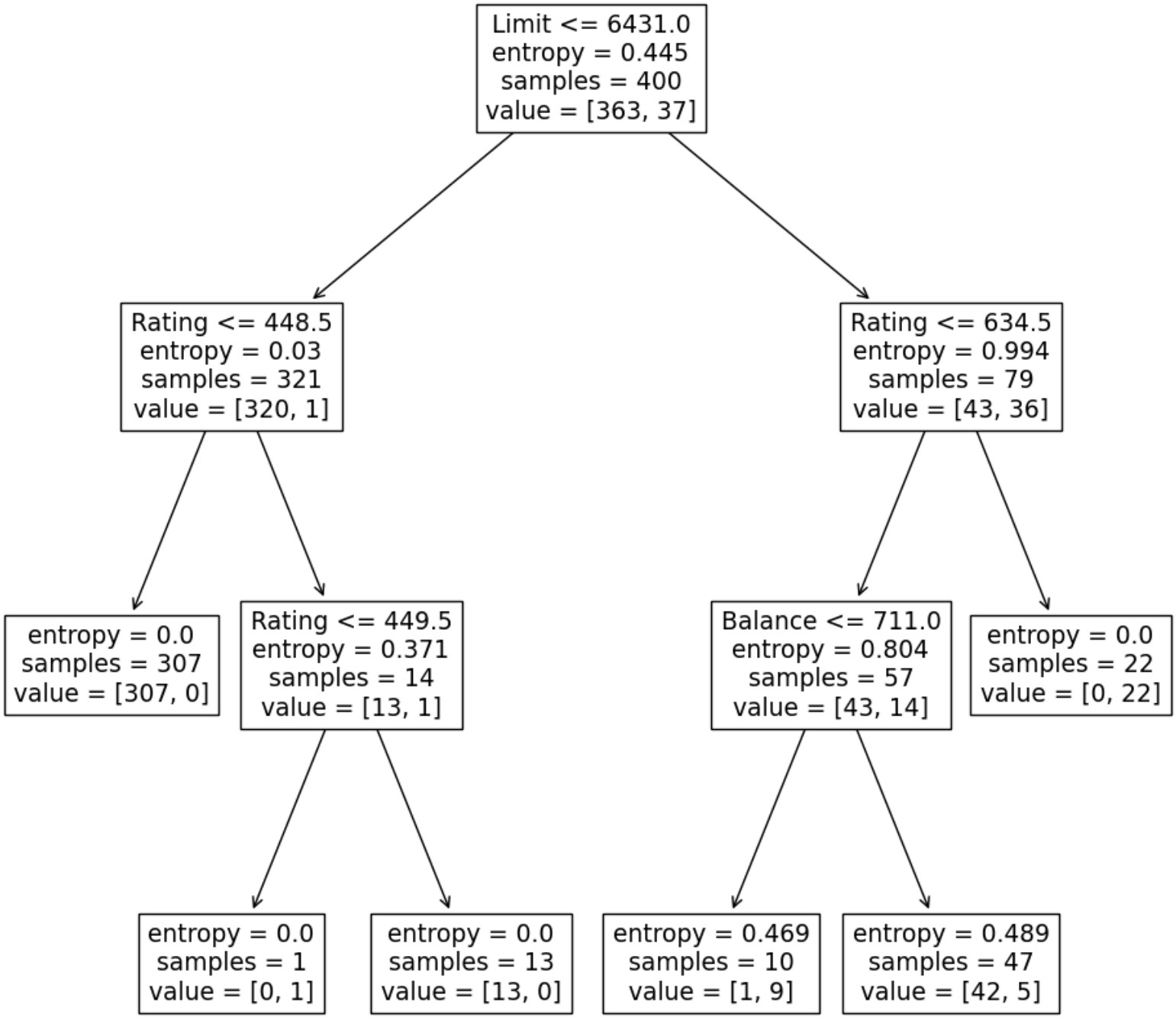
Out[36]:
```
0.985
```

In [37]:
```
resid_dev = np.sum(log_loss(High, clf.predict_proba(X)))
resid_dev
```

Out[37]:
```
0.047946133710538366
```

In [38]:

```
ax = subplots(figsize=(12,12))[1]
plot_tree(clf,
feature_names=feature_names,
ax=ax);
```


Limit <= 6431.0
entropy = 0.445
samples = 400
value = [363, 37]
Rating <= 448.5
entropy = 0.03
samples = 321
value = [320, 1]
Rating <= 634.5
entropy = 0.994
samples = 79
value = [43, 36]
entropy = 0.0
samples = 307
value = [307, 0]
Rating <= 449.5
entropy = 0.371
samples = 14
value = [13, 1]
Balance <= 711.0
entropy = 0.804
samples = 57
value = [43, 14]
entropy = 0.0
samples = 22
value = [0, 22]
entropy = 0.0
samples = 1
value = [0, 1]
entropy = 0.0
samples = 13
value = [13, 0]
entropy = 0.469
samples = 10
value = [1, 9]
entropy = 0.489
samples = 47
value = [42, 5]

In [39]:
```
print(export_text(clf,
feature_names=feature_names,
show_weights=True))
```

```
|--- Limit <= 6431.00
|   |--- Rating <= 448.50
|   |   |--- weights: [307.00, 0.00] class: No
|   |--- Rating >  448.50
|   |   |--- Rating <= 449.50
|   |   |   |--- weights: [0.00, 1.00] class: Yes
|   |   |--- Rating >  449.50
|   |   |   |--- weights: [13.00, 0.00] class: No
|--- Limit >  6431.00
|   |--- Rating <= 634.50
|   |   |--- Balance <= 711.00
|   |   |   |--- weights: [1.00, 9.00] class: Yes
|   |   |--- Balance >  711.00
|   |   |   |--- weights: [42.00, 5.00] class: No
|   |--- Rating >  634.50
|   |   |--- weights: [0.00, 22.00] class: Yes
```

In [40]:
```
validation = skm.ShuffleSplit(n_splits=1,
test_size=200,
random_state=0)
results = skm.cross_validate(clf,
D,
High,
cv=validation)
results['test_score']
```

Out[40]:
```
array([0.965])
```

In [41]:
```
(X_train,
X_test,
High_train,
High_test) = skm.train_test_split(X,
High,
test_size=0.5,
random_state=0)
```

In [42]:
```
clf = DTC(criterion='entropy', random_state=0)
clf.fit(X_train, High_train)
accuracy_score(High_test, clf.predict(X_test))
```

Out[42]: 0.95

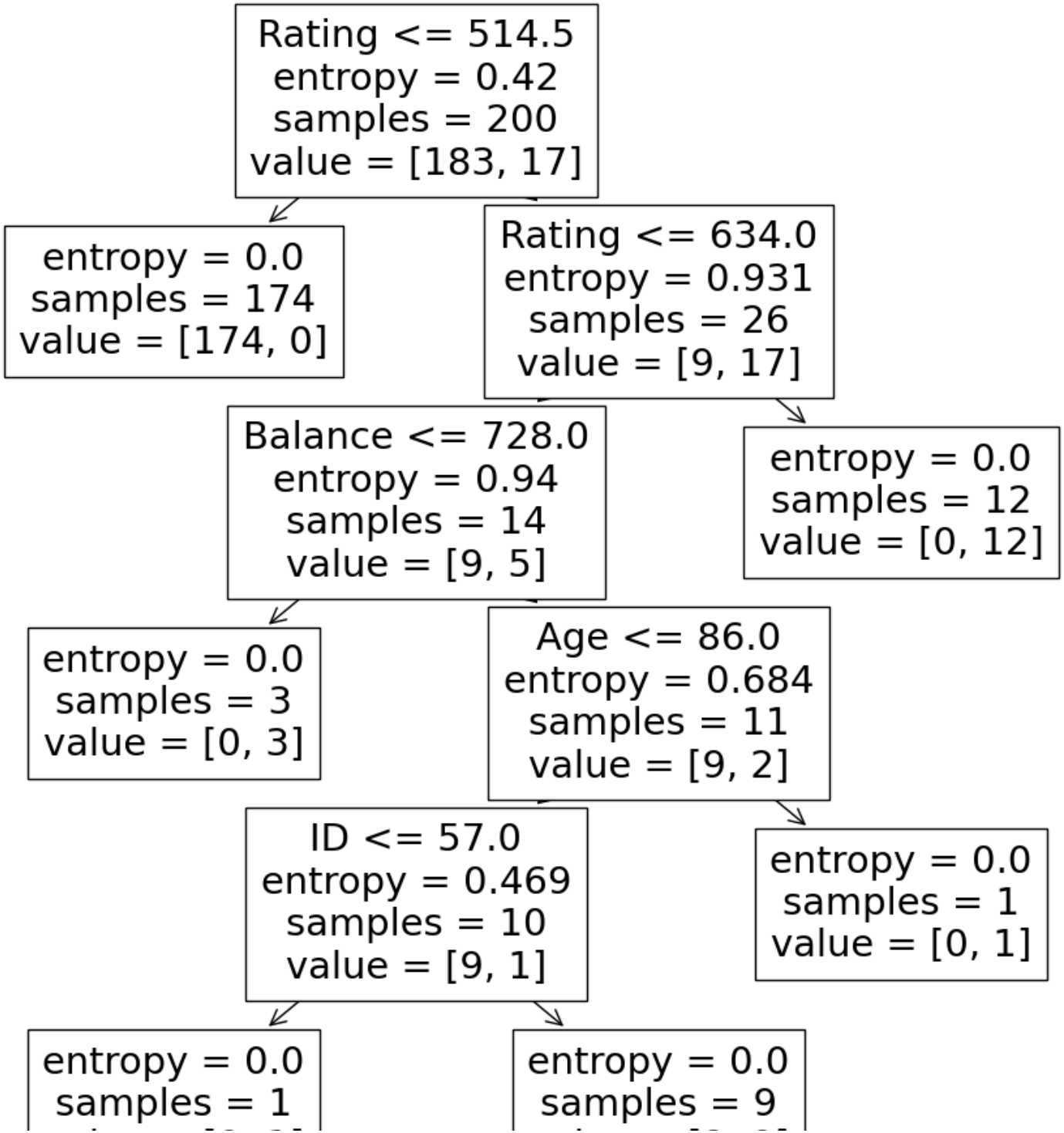
In [43]:
```
ccp_path = clf.cost_complexity_pruning_path(X_train, High_train)
kfold = skm.KFold(10,
random_state=1,
shuffle=True)
```

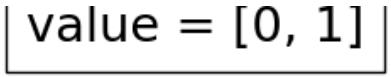
In [44]:
```
grid = skm.GridSearchCV(clf,
{'ccp_alpha': ccp_path.ccp_alphas},
refit=True,cv=kfold,
scoring='accuracy')
grid.fit(X_train, High_train)
grid.best_score_
```

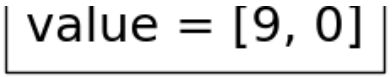
Out[44]: 0.975

In [45]:

```
ax = subplots(figsize=(12, 12))[1]
best_ = grid.best_estimator_
plot_tree(best_,
feature_names=feature_names,
ax=ax);
```


Rating <= 514.5
entropy = 0.42
samples = 200
value = [183, 17]
entropy = 0.0
samples = 174
value = [174, 0]
Rating <= 634.0
entropy = 0.931
samples = 26
value = [9, 17]
Balance <= 728.0
entropy = 0.94
samples = 14
value = [9, 5]
entropy = 0.0
samples = 12
value = [0, 12]
entropy = 0.0
samples = 3
value = [0, 3]
Age <= 86.0
entropy = 0.684
samples = 11
value = [9, 2]
ID <= 57.0
entropy = 0.469
samples = 10
value = [9, 1]
entropy = 0.0
samples = 1
value = [0, 1]
entropy = 0.0
samples = 1
entropy = 0.0
samples = 9





In [46]:
```
best_.tree_.n_leaves
```

Out[46]: 6

In [47]:
```
print(accuracy_score(High_test,
best_.predict(X_test)))
confusion = confusion_table(best_.predict(X_test),
High_test)
confusion
```

0.95

Out[47]:

Truth	No	Yes
Predicted		
No	177	7
Yes	3	13