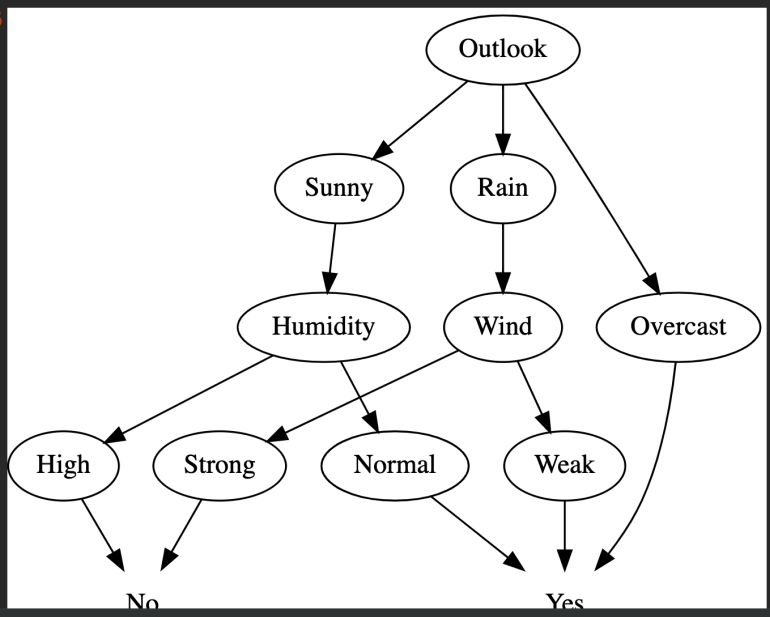
```
#%%
# builds and returns a dictionary decision tree
def build_tree(examples, attributes):
    pure = examples[target_feature].value_counts()
    majority_target_feature_value = pure.idxmax()
    # handles all 3 base cases: if all target values are
    if pure.size == 1 or len(attributes) == 0 or len(exa
        return majority_target_feature_value
    else:
        best_feature = d_best(examples, attributes)
        root = {best_feature : {}} # dictionary of dict
        # remove best_feature from attributes to not te
        attributes = [i for i in attributes if i != best
        unique_values = unique_attribute_values[best_fea
        for value in unique_values:
            # this selects all same value rows of an at
            partition_examples = examples.loc[examples[b
            if partition_examples.empty:
                root[best_feature][value] = majority_tar
            else: # make a subtree by passing same value
                root[best_feature][value] = build_tree(p
        return root

decision_tree = build_tree(example_df, features)
decision_tree

#%%
```

```
162 {'Outlook': {'Sunny': {'Humidity': {'High': 'No', 'Normal': 'Yes'}},
      'Rain': {'Wind': {'Weak': 'Yes', 'Strong': 'No'}},
      'Overcast': 'Yes'}}
```

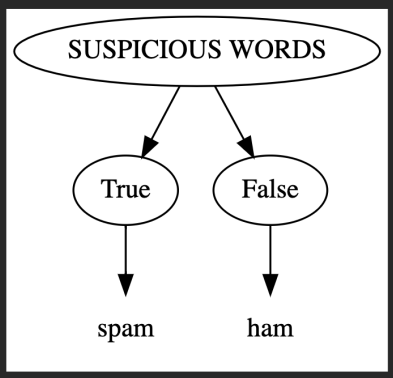
166



```
play_tennis = pd.read_csv("assets/playtennis.csv")
emails = pd.read_csv("assets/emails.csv")
vegetation = pd.read_csv("assets/vegetation.csv")
census = pd.read_csv("assets/census_training.csv")

# TYPE NAME OF CSV FILE TO TRAIN (census default)
example_df = emails
pruning = 0

unique_attribute_values = {}

#%%
# removes columns with all distinct values to avoid over
def remove_distinct_attributes(data_frame):
    for column in data_frame.columns:
        if len(data_frame[column].unique()) == len(data_
            del data_frame[column]
    return data_frame


example_df = remove_distinct_attributes(example_df)

features = example_df.columns.values.tolist()
target_feature = features[-1]
features.pop() # remove target feature
# add feature to dictionary as a key and unique values
for feature in features:
    unique_attribute_values[feature] = pd.value_counts(
```

```
172 {'SUSPICIOUS WORDS': {True: 'spam', False: 'ham'}}
```

176



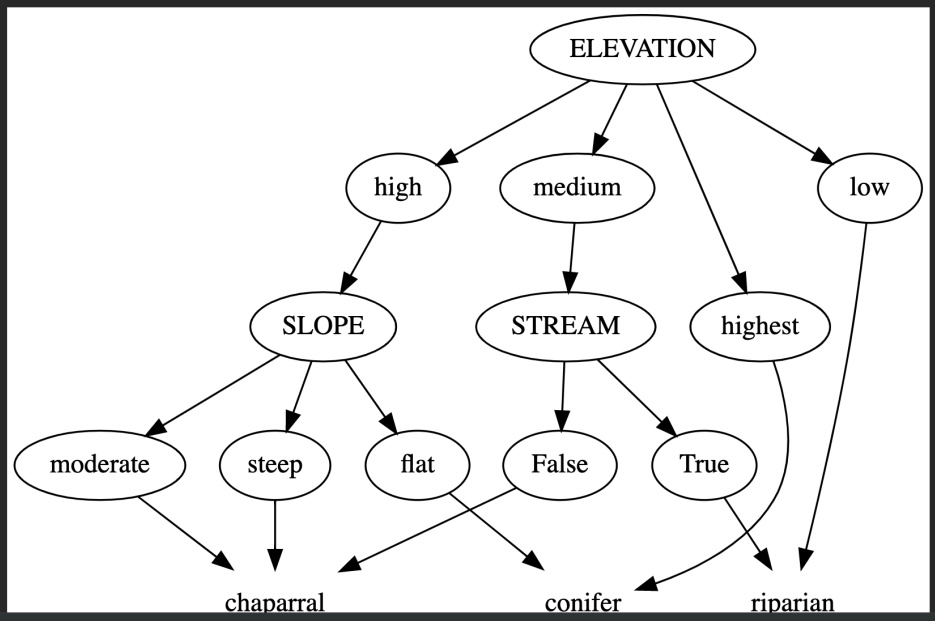
MLA2 〉 ID3.ipynb

id3.py | ID3.ipynb

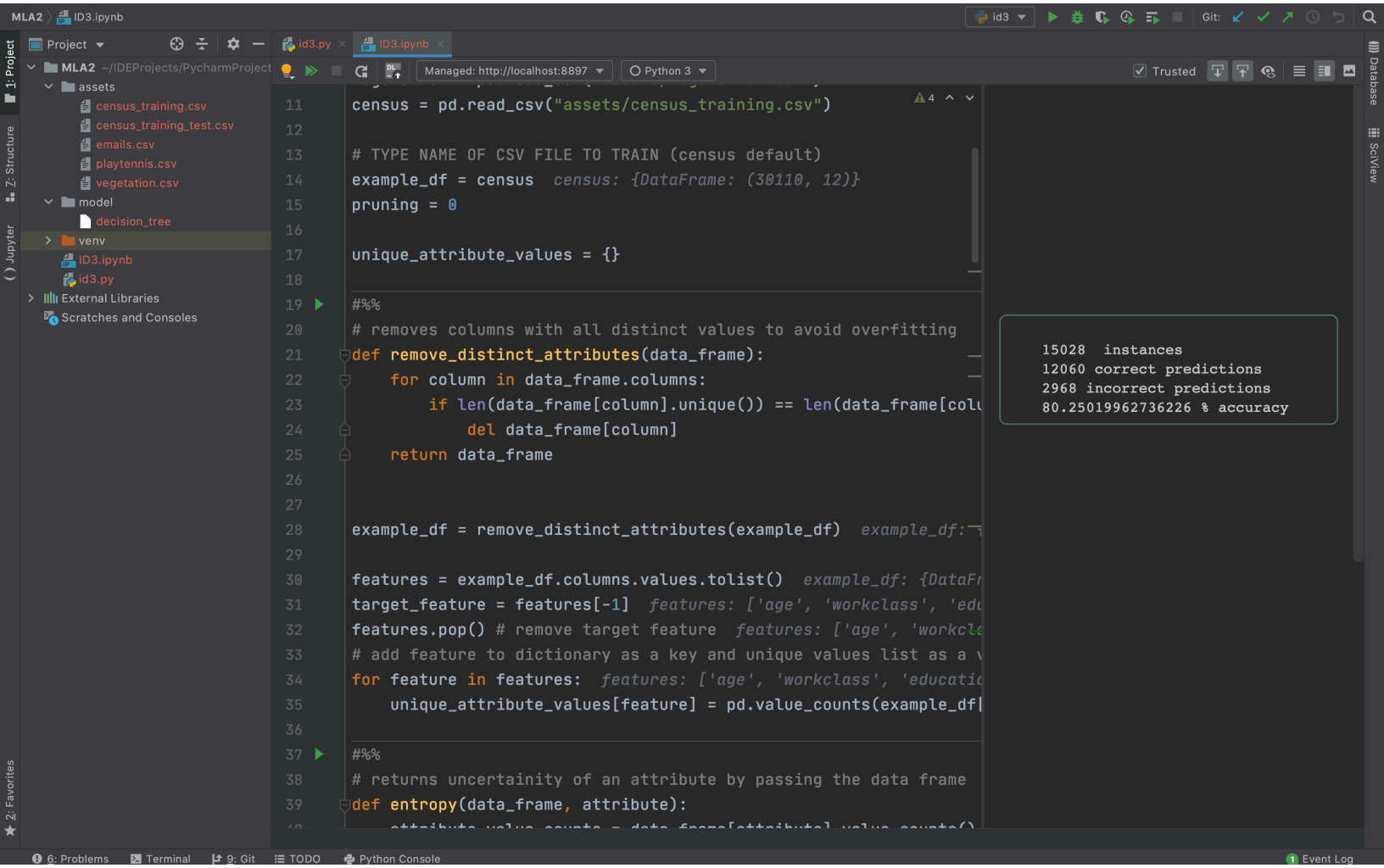
Managed: http://localhost:8786 | Python 3 | Trusted

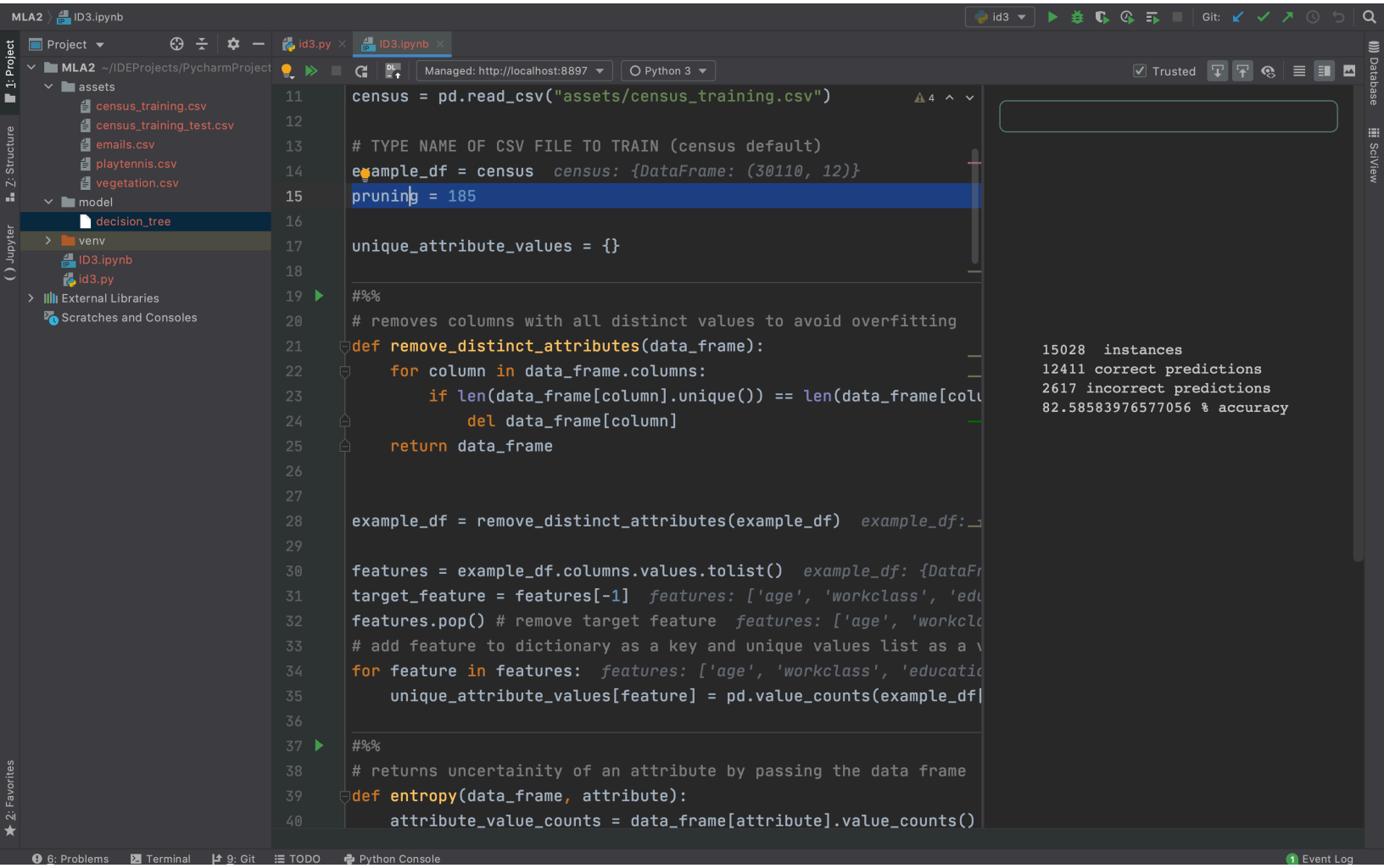
```
# returns uncertainity of an
def entropy(data_frame, attribute):
    attribute_value_counts = data_fr
    set_count = len(data_frame)
    entropy = 0
    for value_count in attribute_val
        probability = value_count/se
        entropy += -probability * np
    return entropy

#%%
# returns information of gain of an
def information_gain(data_frame, att
    value_counts = data_frame[attrib
    target_feature_entropy = entropy
    attribute_entropy = 0
    for value in value_counts.index:
        probability = value_counts[v
        attribute_entropy += probabi
    return target_feature_entropy -


#%%
# calculates best classifer to all a
def d_best(data_frame, attributes):
    entropy = -1
    best_gain = None
    for attribute in attributes:
        current_entropy = informatio
        if current_entropy > entropy
```

```
'moderate': 'chaparral',
'flat': 'conifer'}},
'medium': {'STREAM': {True: 'riparian', False: 'chaparral'}},
'highest': 'conifer',
'low': 'riparian'}}
```




6: Problems | Terminal | 9: Git | TODO | Python Console | Event Log


MLA2 ID3.ipynb
id3
Git:
Project
id3.py
ID3.ipynb
MLA2 ~/IDEProjects/PycharmProject
assets
census_training.csv
census_training_test.csv
emails.csv
playtennis.csv
vegetation.csv
model
decision_tree
venv
ID3.ipynb
id3.py
External Libraries
Scratches and Consoles
Managed: http://localhost:8897
Python 3
Trusted
census = pd.read_csv("assets/census_training.csv")
# TYPE NAME OF CSV FILE TO TRAIN (census default)
example_df = census census: {DataFrame: (30110, 12)}
pruning = 0
unique_attribute_values = {}
#%%
# removes columns with all distinct values to avoid overfitting
def remove_distinct_attributes(data_frame):
for column in data_frame.columns:
if len(data_frame[column].unique()) == len(data_frame[colu
del data_frame[column]
return data_frame
example_df = remove_distinct_attributes(example_df) example_df:
features = example_df.columns.values.tolist() example_df: {DataFr
target_feature = features[-1] features: ['age', 'workclass', 'edu
features.pop() # remove target feature features: ['age', 'workcl
# add feature to dictionary as a key and unique values list as a v
for feature in features: features: ['age', 'workclass', 'educatio
unique_attribute_values[feature] = pd.value_counts(example_df
#%%
# returns uncertainity of an attribute by passing the data frame
def entropy(data_frame, attribute):
15028 instances
12060 correct predictions
2968 incorrect predictions
80.25019962736226 % accuracy
1: Project
7: Structure
Jupyter
2: Favorites
Database
SciView
6: Problems
Terminal
9: Git
TODO
Python Console
Event Log


MLA2 〉 ID3.ipynb
id3
Git:
Project
id3.py
ID3.ipynb
MLA2 ~/IDEProjects/PycharmProject
assets
census_training.csv
census_training_test.csv
emails.csv
playtennis.csv
vegetation.csv
model
decision_tree
venv
ID3.ipynb
id3.py
External Libraries
Scratches and Consoles
Managed: http://localhost:8897
Python 3
Trusted
census = pd.read_csv("assets/census_training.csv")
# TYPE NAME OF CSV FILE TO TRAIN (census default)
example_df = census census: {DataFrame: (30110, 12)}
pruning = 185
unique_attribute_values = {}
#%%
# removes columns with all distinct values to avoid overfitting
def remove_distinct_attributes(data_frame):
for column in data_frame.columns:
if len(data_frame[column].unique()) == len(data_frame[colu
del data_frame[column]
return data_frame
example_df = remove_distinct_attributes(example_df) example_df:_
features = example_df.columns.values.tolist() example_df: {DataFr
target_feature = features[-1] features: ['age', 'workclass', 'edu
features.pop() # remove target feature features: ['age', 'workclo
# add feature to dictionary as a key and unique values list as a v
for feature in features: features: ['age', 'workclass', 'educatio
unique_attribute_values[feature] = pd.value_counts(example_df
#%%
# returns uncertainity of an attribute by passing the data frame
def entropy(data_frame, attribute):
attribute_value_counts = data_frame[attribute].value_counts()
15028 instances
12411 correct predictions
2617 incorrect predictions
82.58583976577056 % accuracy
6: Problems
Terminal
9: Git
TODO
Python Console
Event Log
1: Project
7: Structure
Jupyter
2: Favorites
Database
SciView