projet-2

January 15, 2024

0.0.1 Projet Maching Learning : Analyse Prédictive pour les Admissions dans les Écoles Publiques,

BI&A

[23]:
```
!pip install --upgrade scikit-learn matplotlib
```

```
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-
packages (1.3.2)
Requirement already satisfied: matplotlib in /usr/local/lib/python3.10/dist-
packages (3.8.2)
Requirement already satisfied: numpy<2.0,>=1.17.3 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.23.5)
Requirement already satisfied: scipy>=1.5.0 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.11.4)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-
packages (from scikit-learn) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.2.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.2.0)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (4.47.0)
Requirement already satisfied: kiwisolver>=1.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (1.4.5)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (23.2)
Requirement already satisfied: pillow>=8 in /usr/local/lib/python3.10/dist-
packages (from matplotlib) (9.4.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (3.1.1)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.10/dist-packages (from matplotlib) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-
packages (from python-dateutil>=2.7->matplotlib) (1.16.0)
```

[24]:
```
!pip install kmodes
```

```
Requirement already satisfied: kmodes in /usr/local/lib/python3.10/dist-packages
(0.12.2)
Requirement already satisfied: numpy>=1.10.4 in /usr/local/lib/python3.10/dist-
packages (from kmodes) (1.23.5)
Requirement already satisfied: scikit-learn>=0.22.0 in
/usr/local/lib/python3.10/dist-packages (from kmodes) (1.3.2)
Requirement already satisfied: scipy>=0.13.3 in /usr/local/lib/python3.10/dist-
packages (from kmodes) (1.11.4)
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.10/dist-
packages (from kmodes) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from scikit-learn>=0.22.0->kmodes)
(3.2.0)
```

[25]:
```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in /usr/local/lib/python3.10/dist-
packages (2.15.0)
Requirement already satisfied: absl-py>=1.0.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.4.0)
Requirement already satisfied: astunparse>=1.6.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=23.5.26 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (23.5.26)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.5.4)
Requirement already satisfied: google-pasta>=0.1.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: h5py>=2.9.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (3.9.0)
Requirement already satisfied: libclang>=13.0.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (16.0.6)
Requirement already satisfied: ml-dtypes~=0.2.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.2.0)
Requirement already satisfied: numpy<2.0.0,>=1.23.5 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.23.5)
Requirement already satisfied: opt-einsum>=2.3.2 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (3.3.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (23.2)
Requirement already satisfied:
protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.4,!=4.21.5,<5.0.0dev,>=3.20.3
in /usr/local/lib/python3.10/dist-packages (from tensorflow) (3.20.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (67.7.2)
Requirement already satisfied: six>=1.12.0 in /usr/local/lib/python3.10/dist-
packages (from tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in
```

```
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.4.0)
Requirement already satisfied: typing-extensions>=3.6.6 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (4.5.0)
Requirement already satisfied: wrapt<1.15,>=1.11.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.14.1)
Requirement already satisfied: tensorflow-io-gcs-filesystem>=0.23.1 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (0.35.0)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (1.60.0)
Requirement already satisfied: tensorboard<2.16,>=2.15 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.1)
Requirement already satisfied: tensorflow-estimator<2.16,>=2.15.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: keras<2.16,>=2.15.0 in
/usr/local/lib/python3.10/dist-packages (from tensorflow) (2.15.0)
Requirement already satisfied: wheel<1.0,>=0.23.0 in
/usr/local/lib/python3.10/dist-packages (from astunparse>=1.6.0->tensorflow)
(0.42.0)
Requirement already satisfied: google-auth<3,>=1.6.3 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (2.17.3)
Requirement already satisfied: google-auth-oauthlib<2,>=0.5 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (1.2.0)
Requirement already satisfied: markdown>=2.6.8 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (3.5.1)
Requirement already satisfied: requests<3,>=2.21.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (2.31.0)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in
/usr/local/lib/python3.10/dist-packages (from
tensorboard<2.16,>=2.15->tensorflow) (3.0.1)
Requirement already satisfied: cachetools<6.0,>=2.0.0 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow) (5.3.2)
Requirement already satisfied: pyasn1-modules>=0.2.1 in
/usr/local/lib/python3.10/dist-packages (from google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow) (0.3.0)
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.10/dist-
packages (from google-auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow) (4.9)
Requirement already satisfied: requests-oauthlib>=0.7.0 in
/usr/local/lib/python3.10/dist-packages (from google-auth-
oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow) (1.3.1)
Requirement already satisfied: charset-normalizer<4,>=2 in
```

```
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-
packages (from requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.10/dist-packages (from
requests<3,>=2.21.0->tensorboard<2.16,>=2.15->tensorflow) (2023.11.17)
Requirement already satisfied: MarkupSafe>=2.1.1 in
/usr/local/lib/python3.10/dist-packages (from
werkzeug>=1.0.1->tensorboard<2.16,>=2.15->tensorflow) (2.1.3)
Requirement already satisfied: pyasn1<0.6.0,>=0.4.6 in
/usr/local/lib/python3.10/dist-packages (from pyasn1-modules>=0.2.1->google-
auth<3,>=1.6.3->tensorboard<2.16,>=2.15->tensorflow) (0.5.1)
Requirement already satisfied: oauthlib>=3.0.0 in
/usr/local/lib/python3.10/dist-packages (from requests-oauthlib>=0.7.0->google-
auth-oauthlib<2,>=0.5->tensorboard<2.16,>=2.15->tensorflow) (3.2.2)
```

[26]:

```
! pip install scipy
```

```
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages
(1.11.4)
Requirement already satisfied: numpy<1.28.0,>=1.21.6 in
/usr/local/lib/python3.10/dist-packages (from scipy) (1.23.5)
```

###Projet Machine Learning BI&A

0.0.2 Outline :

1. Exploration des donnees
2. Visualisation
3. codage des donnees
4. Entrainement des modeles Tunning des parametres et evaluation

Librairies :

[27]:

```
import pandas as pd
import numpy as np

# plotting les donnees
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.cluster.hierarchy import dendrogram, linkage
# Preprocessing data
from sklearn.preprocessing import OneHotEncoder
from sklearn.model_selection import train_test_split
from sklearn.decomposition import PCA
```

```python
from sklearn.preprocessing import StandardScaler


from sklearn.utils.class_weight import compute_sample_weight

#Les algorithmes d'apprentissage :
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.multiclass import OneVsRestClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import MultinomialNB
# Clustering des donnees
from kmodes.kmodes import KModes

from sklearn.cluster import AgglomerativeClustering
# MLP
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.layers import  Activation

#les metriques d'evaluation :
from sklearn.metrics import confusion_matrix
from sklearn.metrics import RocCurveDisplay
from sklearn.metrics import accuracy_score, recall_score,␣
↪precision_score,roc_auc_score,classification_report
from sklearn.metrics import silhouette_score
```

[28]:
```python
import pandas as pd

data = pd.read_csv('nursery.csv')
data
```

[28]:
```
          parents   has_nurs      form children     housing     finance  \
0           usual     proper  complete        1  convenient  convenient
1           usual     proper  complete        1  convenient  convenient
2           usual     proper  complete        1  convenient  convenient
3           usual     proper  complete        1  convenient  convenient
4           usual     proper  complete        1  convenient  convenient
…             …          …        …        …        …           …
12955  great_pret  very_crit    foster     more    critical      inconv
12956  great_pret  very_crit    foster     more    critical      inconv
12957  great_pret  very_crit    foster     more    critical      inconv
12958  great_pret  very_crit    foster     more    critical      inconv
12959  great_pret  very_crit    foster     more    critical      inconv

             social       health final evaluation
0           nonprob  recommended        recommend
```

```
1            nonprob     priority         priority
2            nonprob    not_recom        not_recom
3       slightly_prob  recommended        recommend
4       slightly_prob     priority         priority
…                 …            …                …
12955   slightly_prob     priority       spec_prior
12956   slightly_prob    not_recom        not_recom
12957     problematic  recommended       spec_prior
12958     problematic     priority       spec_prior
12959     problematic    not_recom        not_recom

[12960 rows x 9 columns]
```

[29]:
```python
data.head()
```

[29]:
```
  parents has_nurs      form children     housing     finance         social  \
0   usual   proper  complete        1  convenient  convenient        nonprob
1   usual   proper  complete        1  convenient  convenient        nonprob
2   usual   proper  complete        1  convenient  convenient        nonprob
3   usual   proper  complete        1  convenient  convenient  slightly_prob
4   usual   proper  complete        1  convenient  convenient  slightly_prob

        health final evaluation
0  recommended        recommend
1     priority         priority
2    not_recom        not_recom
3  recommended        recommend
4     priority         priority
```

[30]:
```python
#dimensions : nombre de lignes, nombre de colonnes :
print(data.shape)
```

```
(12960, 9)
```

[31]:
```python
# énumération des colonnes :
print(data.columns)
```

```
Index(['parents', 'has_nurs', 'form', 'children', 'housing', 'finance',
       'social', 'health', 'final evaluation'],
      dtype='object')
```

[32]:
```python
#type de chaque colonne :
print(data.dtypes)
```

```
parents             object
has_nurs            object
form                object
children            object
```

```
housing            object
finance            object
social             object
health             object
final evaluation   object
dtype: object
```

[33]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12960 entries, 0 to 12959
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   parents           12960 non-null  object
 1   has_nurs          12960 non-null  object
 2   form              12960 non-null  object
 3   children          12960 non-null  object
 4   housing           12960 non-null  object
 5   finance           12960 non-null  object
 6   social            12960 non-null  object
 7   health            12960 non-null  object
 8   final evaluation  12960 non-null  object
dtypes: object(9)
memory usage: 911.4+ KB
```

[34]:
```
data.describe()
```

[34]:

	parents	has_nurs	form	children	housing	finance	social
count	12960	12960	12960	12960	12960	12960	12960
unique	3	5	4	4	3	2	3
top	usual	proper	complete	1	convenient	convenient	nonprob
freq	4320	2592	3240	3240	4320	6480	4320

	health	final evaluation
count	12960	12960
unique	3	4
top	recommended	not_recom
freq	4320	4320

[35]:
```
data.columns
```

[35]:
```
Index(['parents', 'has_nurs', 'form', 'children', 'housing', 'finance',
       'social', 'health', 'final evaluation'],
      dtype='object')
```

[36]:
```python
print("La liste des colonnes :------")
for x in  data.columns :
  print("")
  print("\n -------- ",x,"-----------")
  #print("\n",(data[x].value_counts()*100)/12960)
  print("\n",(data[x].value_counts()))
```

```
La liste des colonnes :------


 --------  parents -----------

 usual          4320
pretentious    4320
great_pret     4320
Name: parents, dtype: int64


 --------  has_nurs -----------

 proper         2592
less_proper    2592
improper       2592
critical       2592
very_crit      2592
Name: has_nurs, dtype: int64


 --------  form -----------

 complete      3240
completed     3240
incomplete    3240
foster        3240
Name: form, dtype: int64


 --------  children -----------

 1       3240
2       3240
3       3240
more    3240
Name: children, dtype: int64


 --------  housing -----------
```

```
 convenient    4320
less_conv     4320
critical      4320
Name: housing, dtype: int64


 --------  finance -----------

 convenient    6480
inconv        6480
Name: finance, dtype: int64


 --------  social -----------

 nonprob          4320
slightly_prob    4320
problematic      4320
Name: social, dtype: int64


 --------  health -----------

 recommended    4320
priority       4320
not_recom      4320
Name: health, dtype: int64


 --------  final evaluation -----------

 not_recom     4320
priority      4266
spec_prior    4044
recommend      330
Name: final evaluation, dtype: int64
```

[37]:
```
data.iloc[:,0:9]
```

[37]:
```
          parents   has_nurs      form children     housing     finance  \
0           usual     proper  complete        1  convenient  convenient
1           usual     proper  complete        1  convenient  convenient
2           usual     proper  complete        1  convenient  convenient
3           usual     proper  complete        1  convenient  convenient
4           usual     proper  complete        1  convenient  convenient
…             …        …       …        …         …          …
```

```
12955  great_pret  very_crit    foster     more    critical      inconv
12956  great_pret  very_crit    foster     more    critical      inconv
12957  great_pret  very_crit    foster     more    critical      inconv
12958  great_pret  very_crit    foster     more    critical      inconv
12959  great_pret  very_crit    foster     more    critical      inconv

              social       health final evaluation
0            nonprob  recommended        recommend
1            nonprob     priority         priority
2            nonprob    not_recom        not_recom
3      slightly_prob  recommended        recommend
4      slightly_prob     priority         priority
…                …          …              …
12955  slightly_prob     priority       spec_prior
12956  slightly_prob    not_recom        not_recom
12957    problematic  recommended       spec_prior
12958    problematic     priority       spec_prior
12959    problematic    not_recom        not_recom

[12960 rows x 9 columns]
```

- L'evaluation final en fonction des parents :

[38]:
```
parents_pret = data.loc[data['parents']== "great_pret",:]
```

[39]:
```
(parents_pret['final evaluation'].value_counts()*100)/4320
# On remarque 46.8% des candidats possedent  plus haut niveau de priorité,
#indiquant que le candidat devrait bénéficier d'une considération spéciale et␣
↪d'une acceptation
```

[39]:
```
spec_prior    46.805556
not_recom     33.333333
priority      19.861111
Name: final evaluation, dtype: float64
```

[40]:
```
parents_usual = data.loc[data['parents']== "usual",:]
```

[41]:
```
(parents_usual['final evaluation'].value_counts()*100)/4320
```

[41]:
```
priority      44.537037
not_recom     33.333333
spec_prior    17.546296
recommend      4.583333
Name: final evaluation, dtype: float64
```

[42]:
```
parents_pretentious = data.loc[data['parents']=="pretentious",:]
(parents_pretentious['final evaluation'].value_counts()*100)/4320
```

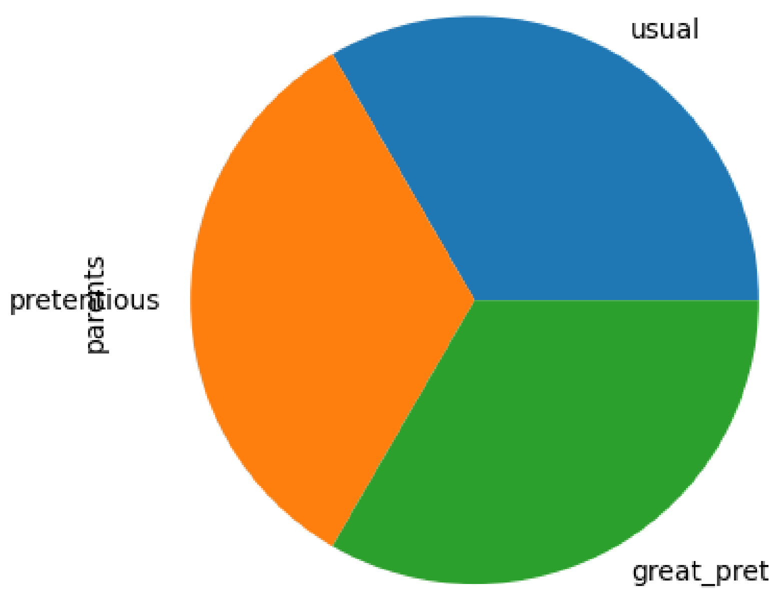
[42]:
```
priority      34.351852
not_recom     33.333333
spec_prior    29.259259
recommend      3.055556
Name: final evaluation, dtype: float64
```

[43]:
```python
print(pd.crosstab(data['parents'],data['final evaluation']))
```

```
final evaluation  not_recom  priority  recommend  spec_prior
parents
great_pret             1440       858          0        2022
pretentious            1440      1484        132        1264
usual                  1440      1924        198         758
```

[44]:
```python
data['parents'].value_counts().plot.pie(subplots= True)
```

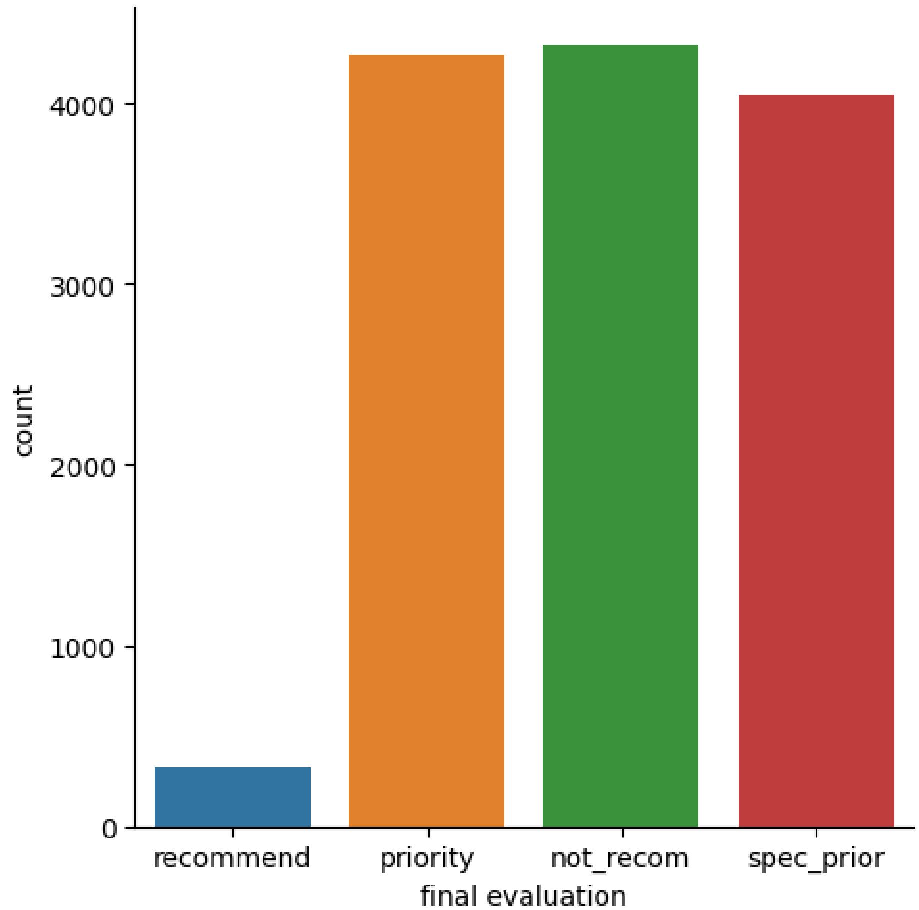
[44]:
```
array([<Axes: ylabel='parents'>], dtype=object)
```



0.0.3 Visualisation des donnees

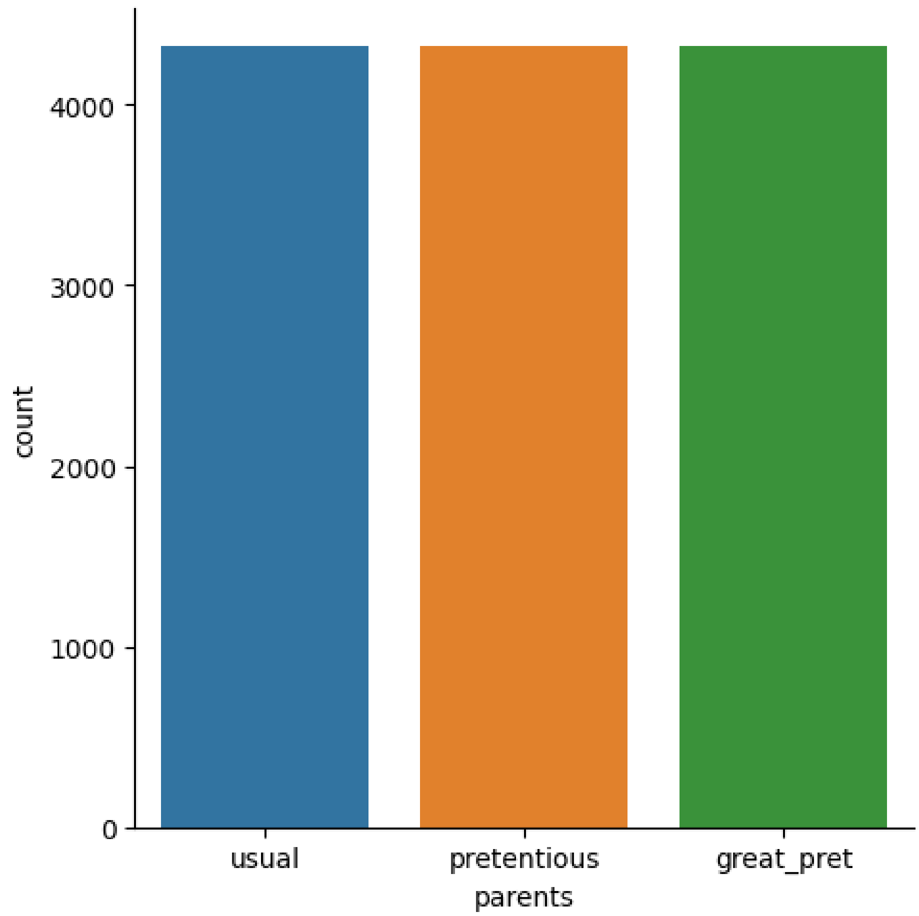
[45]:
```python
#les classes de la variable cible "final evaluation"
sns.catplot(data=data, x="final evaluation", kind="count")
```

[45]: <seaborn.axisgrid.FacetGrid at 0x7f1b54c162f0>



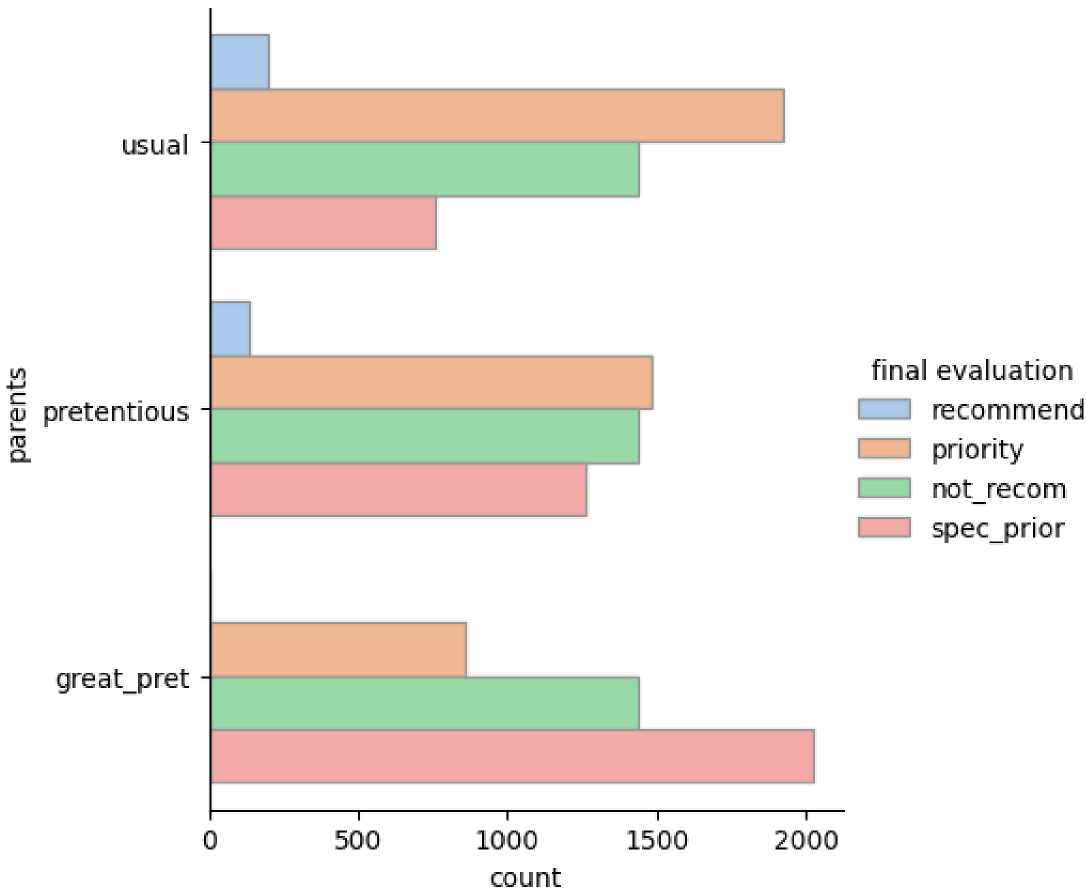
[46]:
```
#bar plot d'occupation des parents
sns.catplot(data=data, x="parents", kind="count")
```

[46]: <seaborn.axisgrid.FacetGrid at 0x7f1ac45968c0>



[47]:
```
#Répartition des évaluations finales en fonction des professions des parents
sns.catplot(
    data=data, y="parents", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[47]: <seaborn.axisgrid.FacetGrid at 0x7f1ac21cc490>

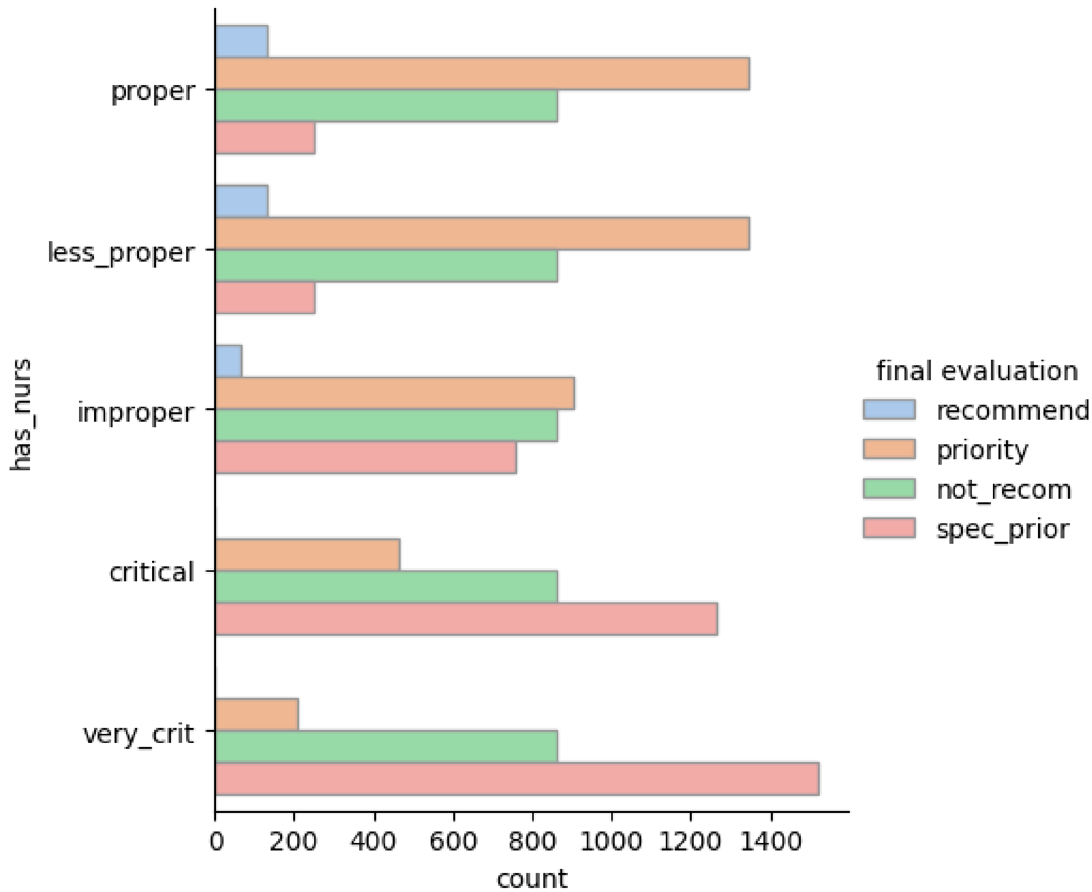


[48]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12960 entries, 0 to 12959
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype 
---  ------            --------------  ----- 
 0   parents           12960 non-null  object
 1   has_nurs          12960 non-null  object
 2   form              12960 non-null  object
 3   children          12960 non-null  object
 4   housing           12960 non-null  object
 5   finance           12960 non-null  object
 6   social            12960 non-null  object
 7   health            12960 non-null  object
 8   final evaluation  12960 non-null  object
dtypes: object(9)
memory usage: 911.4+ KB
```

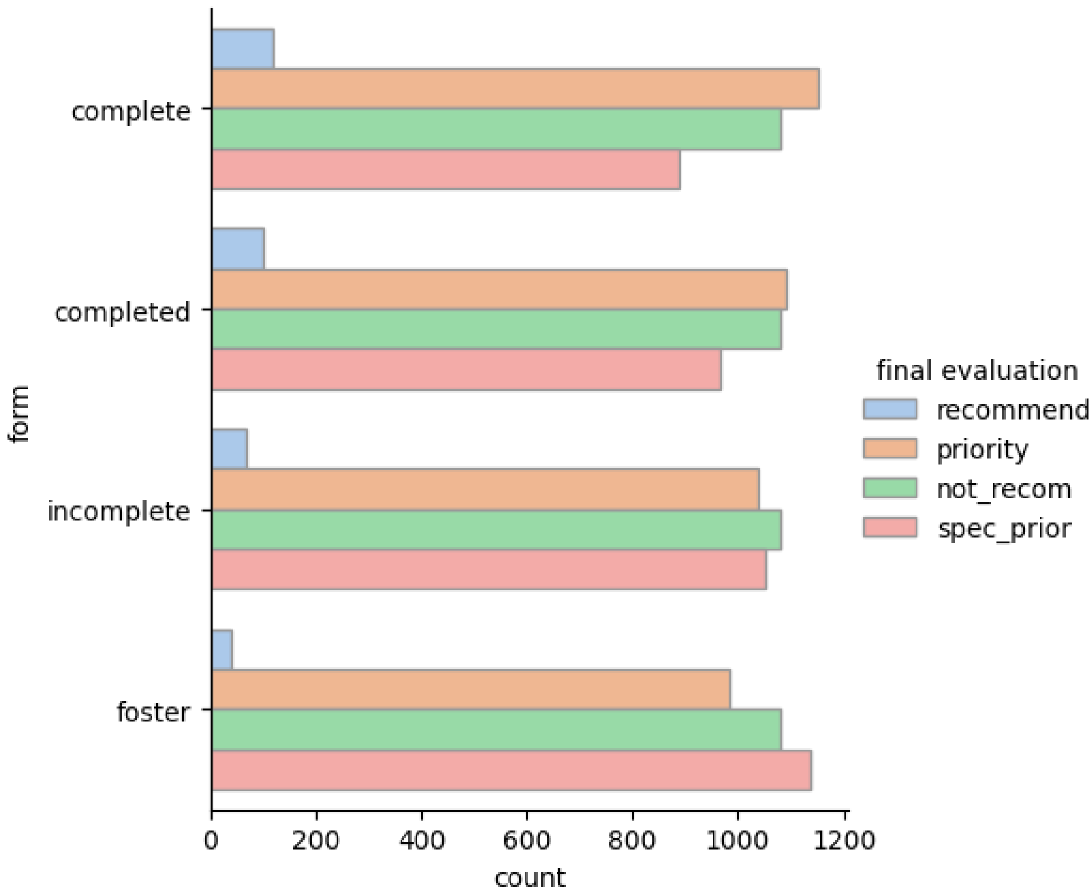
[49]:
```
#Répartition des évaluations finales en fonction de garderie de l'enfant
sns.catplot(
    data=data, y="has_nurs", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[49]: <seaborn.axisgrid.FacetGrid at 0x7f1ac223f190>



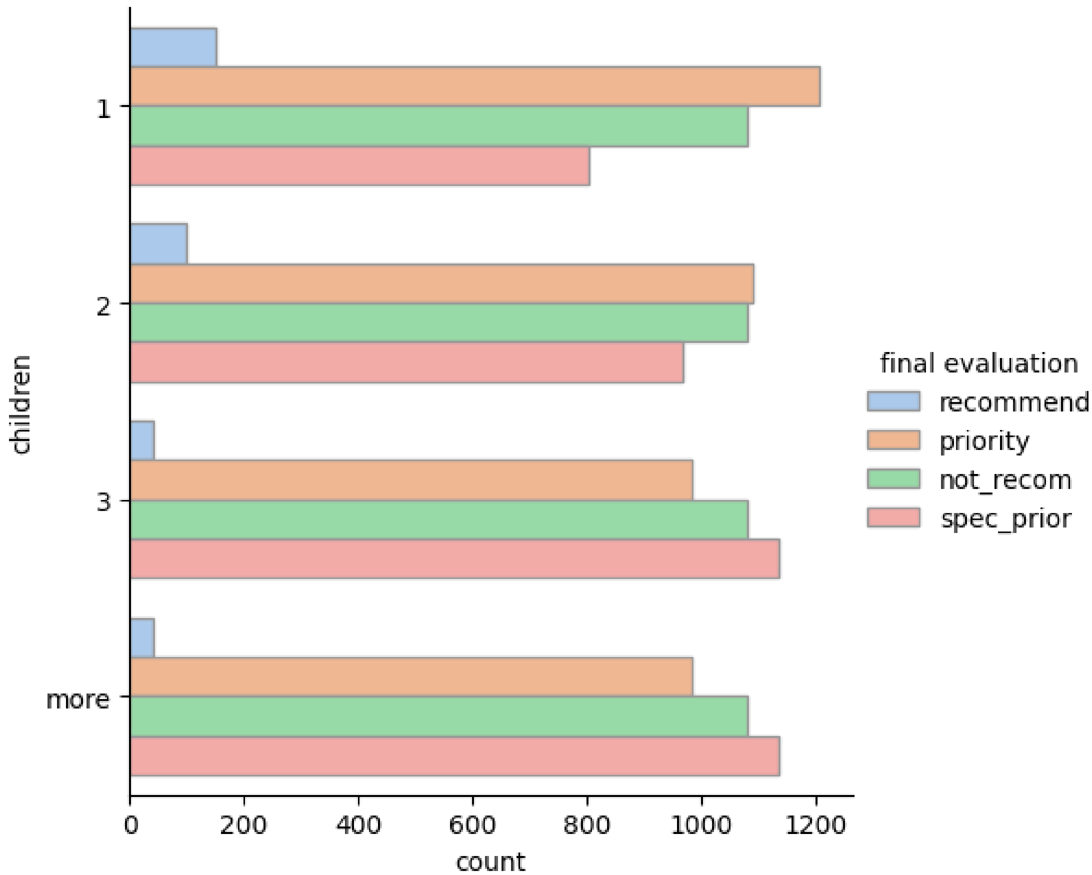
[50]:
```
#Répartition des évaluations finales en fonction de la structure de la famille
sns.catplot(
    data=data, y="form", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[50]: <seaborn.axisgrid.FacetGrid at 0x7f1ac2150d00>



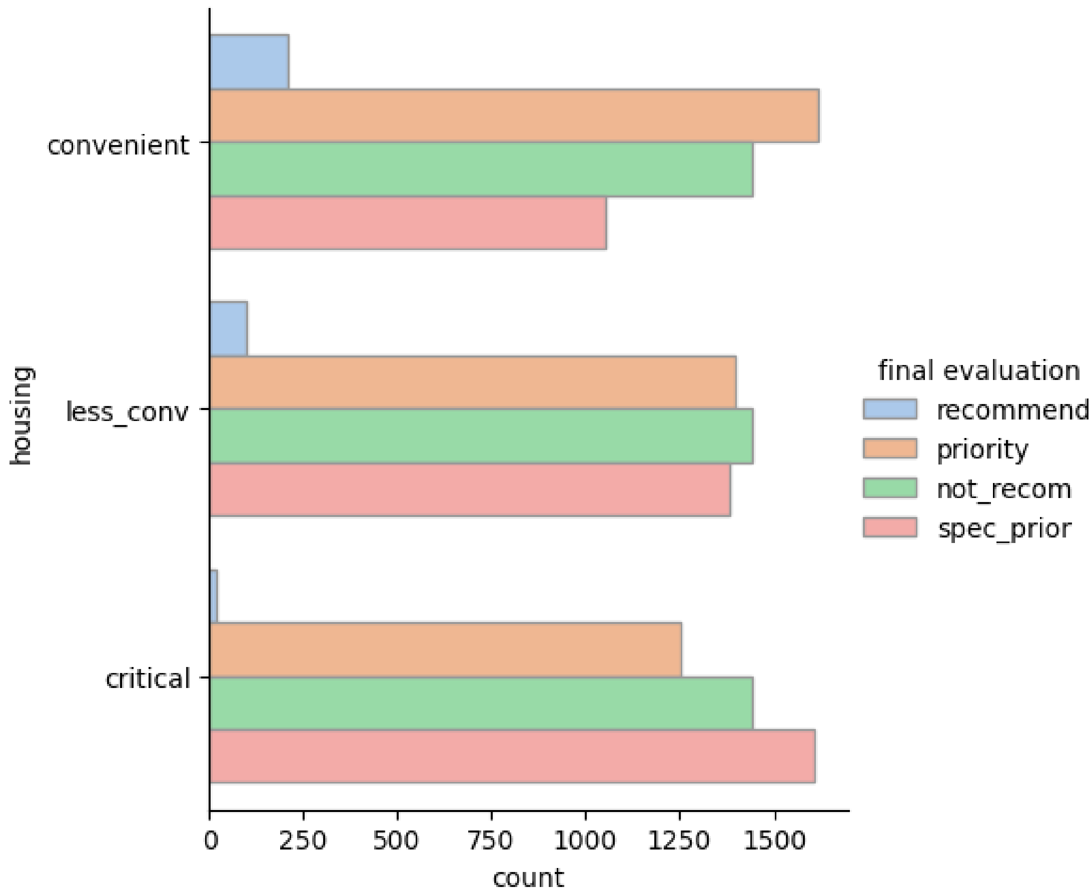
[51]:
```
#Répartition des évaluations finales en fonction de nombre d'enfants
sns.catplot(
    data=data, y="children", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[51]: <seaborn.axisgrid.FacetGrid at 0x7f1ac2151f00>



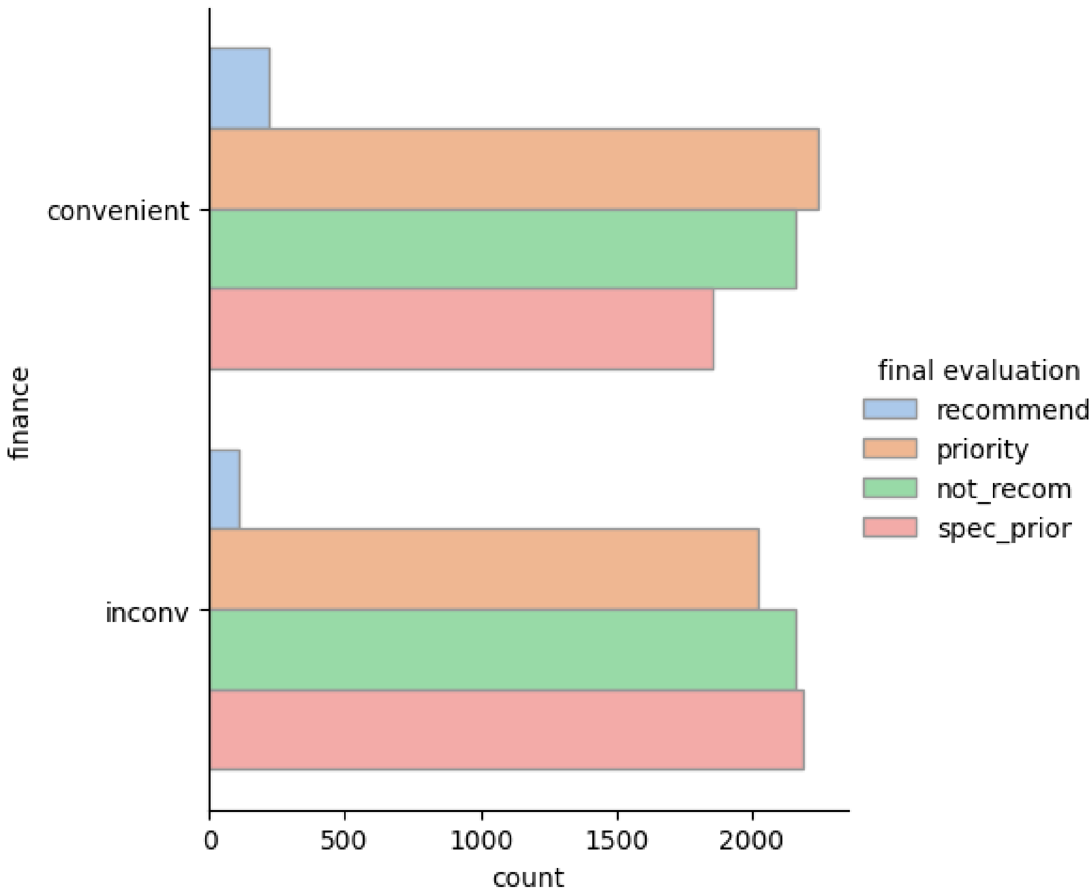
[52]:
```
#Répartition des évaluations finales en fonction des conditions de logement
sns.catplot(
    data=data, y="housing", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[52]: <seaborn.axisgrid.FacetGrid at 0x7f1ac20a40a0>



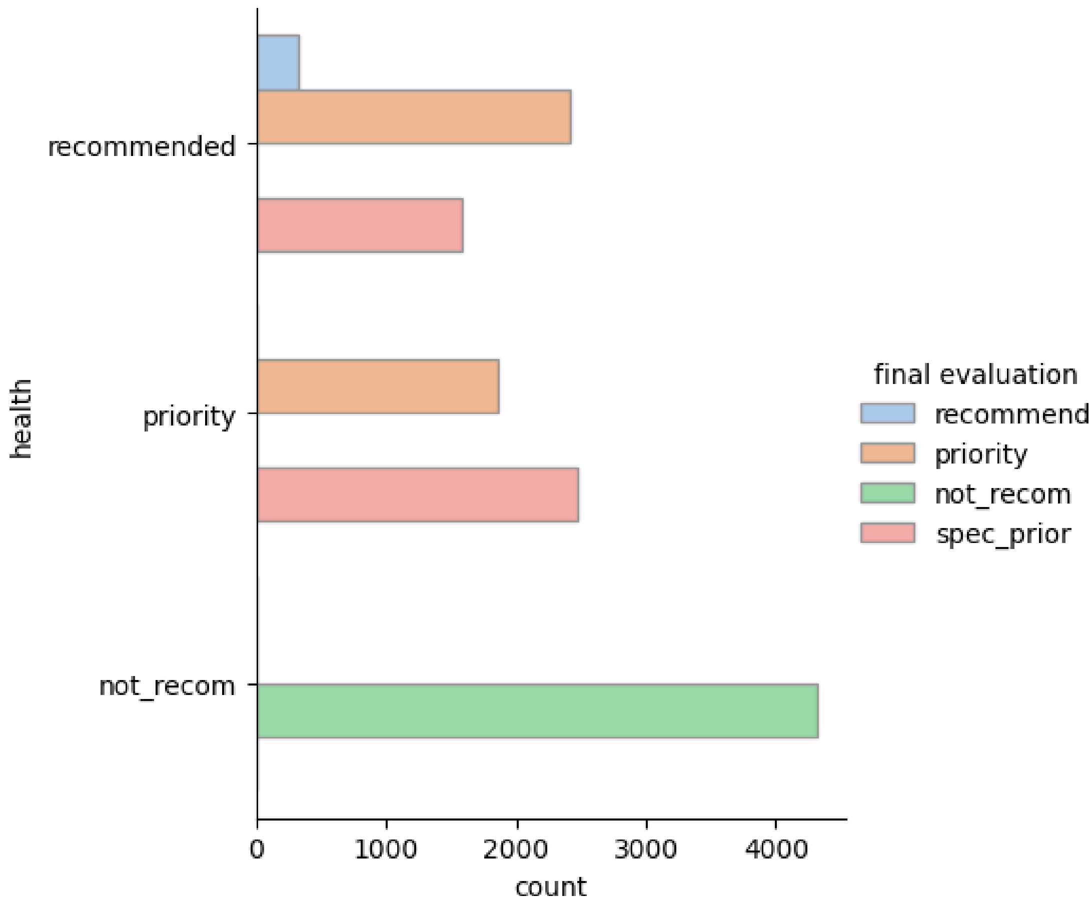
[53]:
```python
#Répartition des évaluations finales en fonction de la situation financiere
sns.catplot(
    data=data, y="finance", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[53]: <seaborn.axisgrid.FacetGrid at 0x7f1ac2073fd0>



[54]:
```
#Répartition des évaluations finales en fonction des conditions sanitaires
sns.catplot(
    data=data, y="health", hue="final evaluation", kind="count",
    palette="pastel", edgecolor=".6",
)
```

[54]: <seaborn.axisgrid.FacetGrid at 0x7f1ac21764a0>



0.0.4 Codage des donnees

[55]:
```python
# Extraction et separation du varaible cible des autres variables
X = data.iloc[:,0:8]
y= data['final evaluation']
y
```

[55]:
```
0          recommend
1           priority
2          not_recom
3          recommend
4           priority
             …
12955     spec_prior
12956      not_recom
12957     spec_prior
12958     spec_prior
12959      not_recom
```

```
Name: final evaluation, Length: 12960, dtype: object
```

###Clustering des donnees

###KModes

[56]:
```
# On pose un random seed pour controler l'aleatoire
random_seed = 42
np.random.seed(seed = 42)
"""
Puisque l'ensemble des donnees est largement sufiisant Alors,
On effectue un split random  pour la construction des ensembles d'entrainement␣
 ↪et de test.
"""

X_train, X_test, y_train, y_test = train_test_split ( X, y, test_size=0.3,␣
 ↪random_state= random_seed)
```

[57]:
```
# Codage des donnees
X_train = pd.get_dummies(X_train)
y_train = pd.get_dummies(y_train)
```

[58]:
```
"""
L'argument init ='Huang' represente une methode d'initialisation particuliere␣
 ↪pour
les centroids. En tant que cet etape est cruciale et a un impact siginificatif␣
 ↪sur
la construction des groupes.Alors l'idee est d'essayer d'obtenir une␣
 ↪repartition initiale
des centroids qui maximise la diversite dans les clusters , en aidant notre␣
 ↪algorithme
a converger vers une meilleure solution globale "

"""
#initialisation du modele et entrainement
model = KModes(n_clusters=4, init='random', n_init=5, verbose=1)
clusters = model.fit_predict(X_train)

# Afficher les centroids des clusters et attribuer les clusters au DataFrame␣
 ↪d'origine
X_train_clustered = X_train.copy()
X_train_clustered['Cluster'] = clusters
print("Cluster Centroids:")
print(pd.DataFrame(model.cluster_centroids_, columns=X_train.columns))

# Plot a count of points in each cluster
plt.figure(figsize=(8, 6))
```

```python
sns.countplot(x='Cluster', data=X_train_clustered)
plt.title("Distribution of Points in Each Cluster")
plt.show()
```

```
Init: initializing centroids
Init: initializing clusters
Starting iterations…
Run 1, iteration: 1/100, moves: 3027, cost: 61320.0
Run 1, iteration: 2/100, moves: 479, cost: 61320.0
Init: initializing centroids
Init: initializing clusters
Starting iterations…
Run 2, iteration: 1/100, moves: 2838, cost: 61452.0
Run 2, iteration: 2/100, moves: 490, cost: 61452.0
Init: initializing centroids
Init: initializing clusters
Starting iterations…
Run 3, iteration: 1/100, moves: 3355, cost: 61872.0
Run 3, iteration: 2/100, moves: 2001, cost: 60217.0
Run 3, iteration: 3/100, moves: 1166, cost: 60217.0
Init: initializing centroids
Init: initializing clusters
Starting iterations…
Run 4, iteration: 1/100, moves: 3522, cost: 61956.0
Init: initializing centroids
Init: initializing clusters
Starting iterations…
Run 5, iteration: 1/100, moves: 3200, cost: 60004.0
Run 5, iteration: 2/100, moves: 1020, cost: 58986.0
Run 5, iteration: 3/100, moves: 23, cost: 58986.0
Best run was number 5
Cluster Centroids:
   parents_great_pret  parents_pretentious  parents_usual  has_nurs_critical  \
0                   0                    0              0                  0   
1                   0                    0              0                  0   
2                   0                    0              0                  0   
3                   0                    0              0                  0   

   has_nurs_improper  has_nurs_less_proper  has_nurs_proper  \
0                  0                     0                0   
1                  0                     0                0   
2                  0                     0                0   
3                  0                     0                0   

   has_nurs_very_crit  form_complete  form_completed  …  housing_critical  \
0                   0              0               0  …                 0   
1                   0              0               0  …                 0   
```

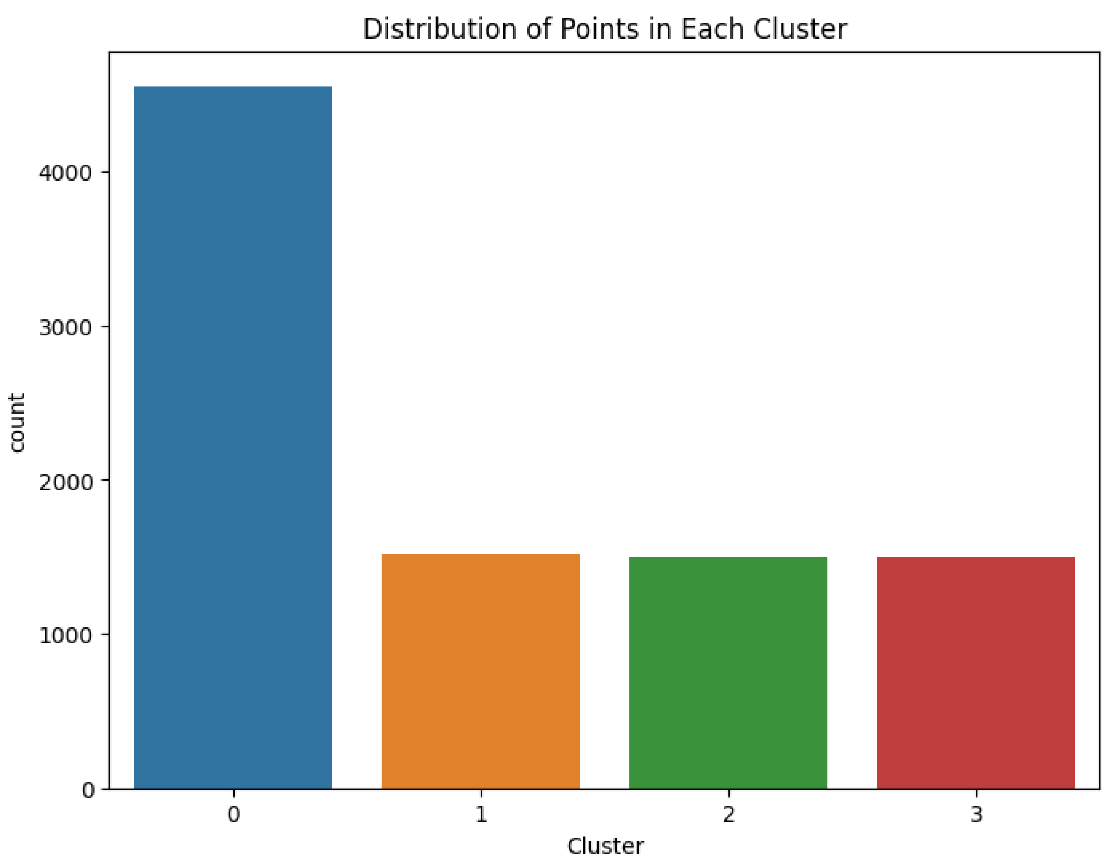
```
2                        0                0               0  ...               1
3                        0                0               0  ...               0

   housing_less_conv  finance_convenient  finance_inconv  social_nonprob  \
0                  0                   1               0               0
1                  1                   0               1               0
2                  0                   0               1               0
3                  0                   0               1               0

   social_problematic  social_slightly_prob  health_not_recom  \
0                   0                     0                 0
1                   0                     0                 0
2                   0                     0                 0
3                   0                     0                 0

   health_priority  health_recommended
0                0                   0
1                0                   0
2                0                   0
3                0                   0

[4 rows x 27 columns]
```



[59]:
```python
print(model.cluster_centroids_)
```

```
[[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0]]
```

[60]:
```python
# Evaluate the clustering using silhouette score
silhouette_avg = silhouette_score(X_train, clusters)
print(f"Silhouette Score: {silhouette_avg}")
```

```
Silhouette Score: 0.06748194668280402
```

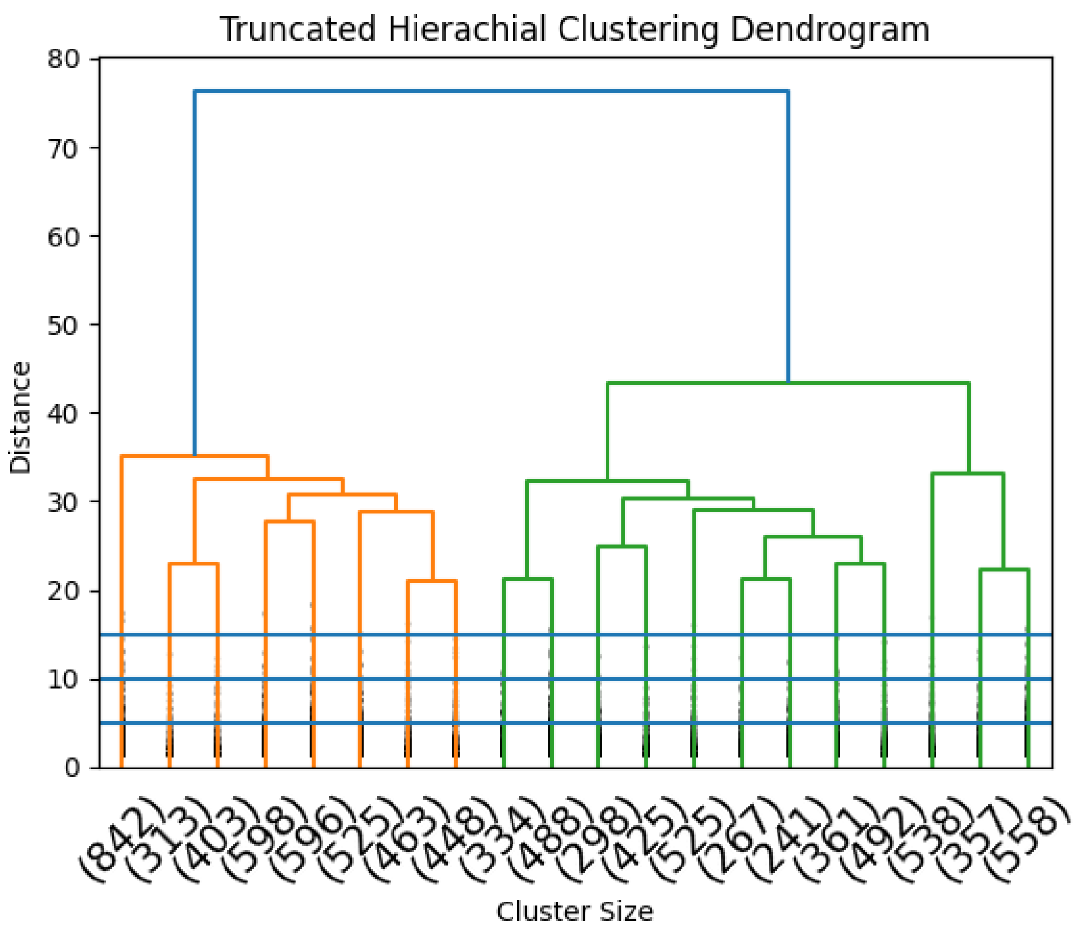
###AgglomerativeClustering

[61]:
```python
"""
Intilisation des parametres du modele , distance "hamming" est "Adéquat
aux donnees categorielles .
"""
Clust = AgglomerativeClustering(n_clusters = 5, linkage= "complete" , affinity␣
↪= 'hamming' )
cluster_labels = Clust.fit_predict(X_train)
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_agglomerative.py:1006:
FutureWarning: Attribute `affinity` was deprecated in version 1.2 and will be
removed in 1.4. Use `metric` instead
  warnings.warn(
```

[62]:
```python
#Construction du dendogramme
z = linkage(X_train,"ward")

#generate dendrogram
dendrogram(z,truncate_mode= "lastp", p =20, leaf_rotation=45,leaf_font_size=15,␣
↪show_contracted=True)
plt.title("Truncated Hierachial Clustering Dendrogram")
plt.xlabel("Cluster Size")
plt.ylabel("Distance")
#divide the cluster
plt.axhline(y=15)
plt.axhline(5)
plt.axhline(10)
plt.show()
```



[63]:
```
#X_train.shape
X_train.shape
```

[63]: (9072, 27)

Entrainement des modeles ,Tunning des parametres et evaluation

[64]:
```
X_test.shape
y_test.shape
```

[64]: (3888,)

Il ya un probleme des classes déséquilibrées dans dans notre cas qui est un défi courant. Donc on va introduire des poids pour essayer de les equilibrer

[65]:
```
class_weights = 'balanced'
```

0.0.5 KNN

[66]:
```python
"""
L'utilsation de l'argument 'distance permet de pondere les points en fonction
de l'inverse de leur distance. Alors, dans ce cas les voisins proches d'un␣
↪point de
requete auront une plus grande influence que les voisins qui sont plus éloignés␣
↪.

"""
knn = KNeighborsClassifier(weights = 'distance')
knn.fit(X_train,y_train)
```

[66]:
```
KNeighborsClassifier(weights='distance')
```

[67]:
```python
# codage des donnnes de test
X_test_cd = pd.get_dummies(X_test)
y_test_cd = pd.get_dummies(y_test)
```

[68]:
```python
# la tache du test
y_pred = knn.predict(X_test_cd)
y_pred
```

[68]:
```
array([[1, 0, 0, 0],
       [0, 1, 0, 0],
       [0, 1, 0, 0],
       …,
       [1, 0, 0, 0],
       [0, 0, 0, 1],
       [0, 0, 0, 1]], dtype=uint8)
```

[69]:
```python
y_test_cd
```

[69]:

	not_recom	priority	recommend	spec_prior
6407	1	0	0	0
6301	0	0	0	1
304	0	1	0	0
12520	0	0	0	1
2417	1	0	0	0
…	…	…	…	…
12346	0	0	0	1
7348	0	0	0	1
12887	1	0	0	0
10228	0	0	0	1
3886	0	0	0	1

[3888 rows x 4 columns]

[70]:
```
# comparaison des dimensions des deux  ensembles
y_test_cd.shape , y_pred.shape
```

[70]: ((3888, 4), (3888, 4))

[71]:
```
#les metriques d'evaluation
accuracy = accuracy_score(y_test_cd, y_pred)

clasreport = classification_report(y_test_cd, y_pred)

print(clasreport)
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1320
           1       0.89      0.88      0.88      1272
           2       0.95      0.35      0.51       106
           3       0.94      0.89      0.91      1190

   micro avg       0.94      0.91      0.93      3888
   macro avg       0.94      0.78      0.83      3888
weighted avg       0.94      0.91      0.92      3888
 samples avg       0.91      0.91      0.91      3888
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1471:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in samples with no predicted labels. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

0.0.6 SVM (One vs Rest)

[72]:
```
# On utilise SVM avec l'approche One vs Rest pour effectuer la classification␣
↪multiclass

ClassSVM = OneVsRestClassifier(SVC(class_weight=class_weights)).fit(X_train,␣
↪y_train)
```

[73]:
```
#On effectue la tache du test
y_pred = ClassSVM.predict(X_test_cd)
```

[74]:
```
# Rapport d'evaluation
print(classification_report(y_test_cd, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1320
```

```
               1       0.92      1.00      0.96      1272
               2       0.99      1.00      1.00       106
               3       1.00      1.00      1.00      1190

       micro avg       0.97      1.00      0.99      3888
       macro avg       0.98      1.00      0.99      3888
    weighted avg       0.97      1.00      0.99      3888
     samples avg       0.99      1.00      0.99      3888
```

[75]:
```
# variable cible du donnees test non-codee
y_test
```

[75]:
```
6407      not_recom
6301     spec_prior
304        priority
12520    spec_prior
2417      not_recom
            …
12346    spec_prior
7348     spec_prior
12887     not_recom
10228    spec_prior
3886     spec_prior
Name: final evaluation, Length: 3888, dtype: object
```

ROC

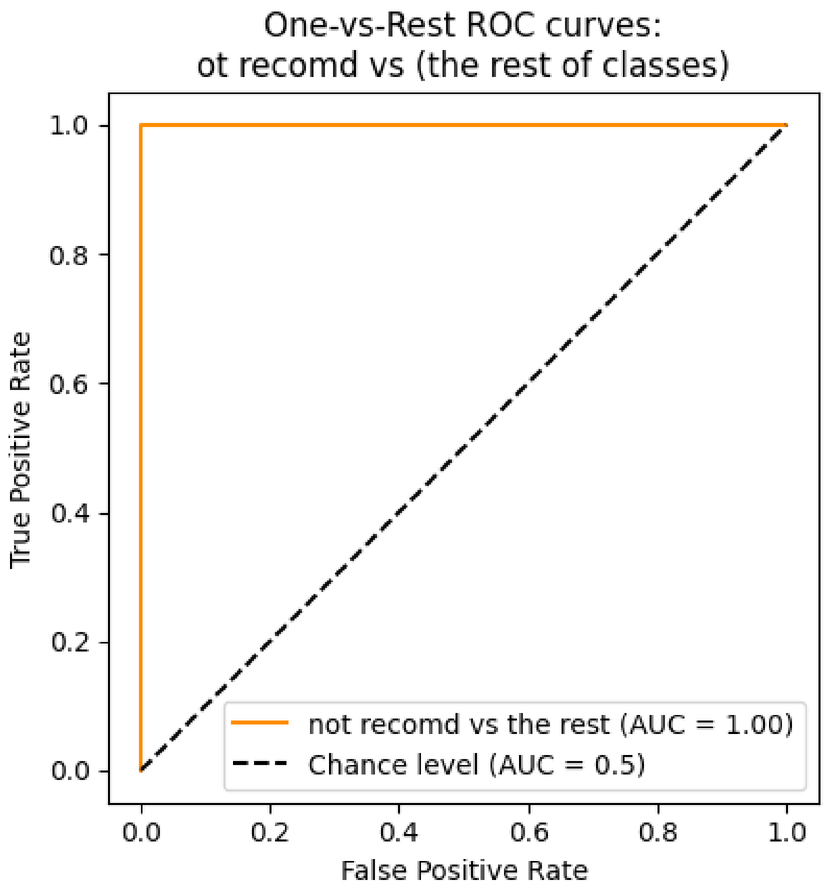
[76]:
```
#variable cible du donnees test codee
y_test_cd
```

[76]:
```
       not_recom  priority  recommend  spec_prior
6407           1         0          0           0
6301           0         0          0           1
304            0         1          0           0
12520          0         0          0           1
2417           1         0          0           0
…            …       …        …         …
12346          0         0          0           1
7348           0         0          0           1
12887          1         0          0           0
10228          0         0          0           1
3886           0         0          0           1

[3888 rows x 4 columns]
```

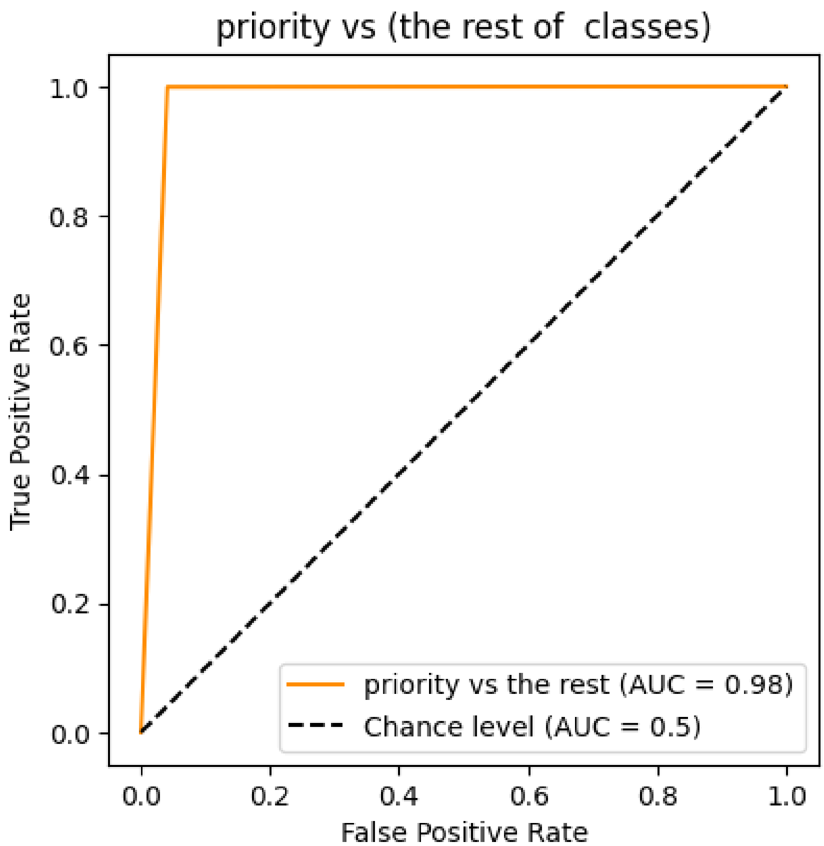
[77]:
```
# plot de courbe ROC pour la classe 'not recomd'
RocCurveDisplay.from_predictions(
```

```
      y_test_cd.iloc[:, 0]
      , y_pred[:, 0],
      name=f"not recomd vs the rest",
      color="darkorange",
      plot_chance_level=True, )
plt.axis("square")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("One-vs-Rest ROC curves:\not recomd vs (the rest of classes)")
plt.legend()
plt.show()
```

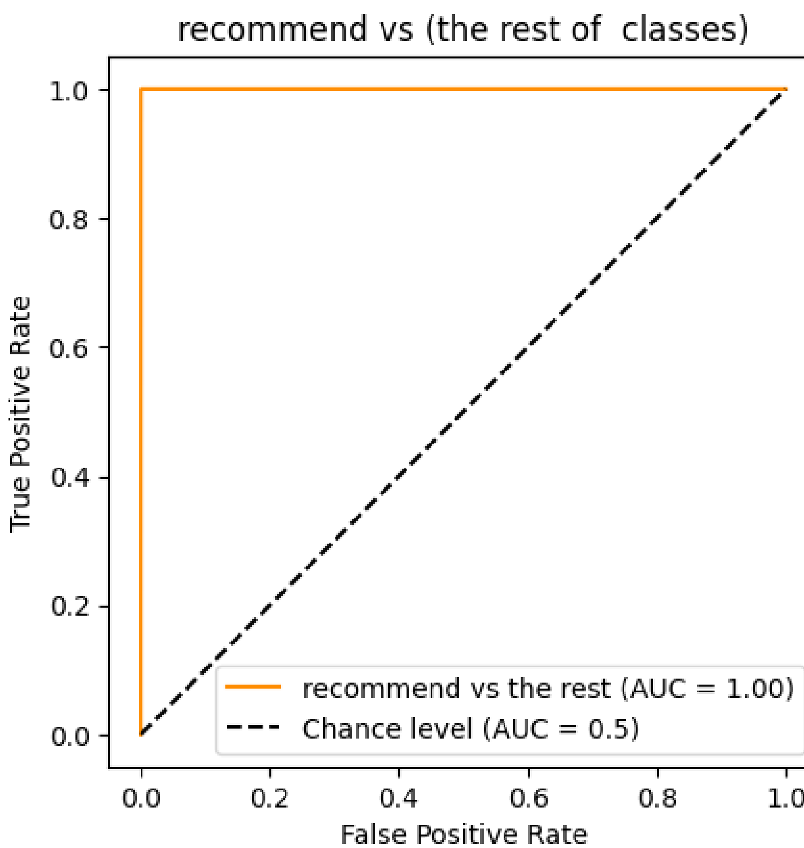


[78]:
```
# plot de courbe ROC pour la classe 'priority'
RocCurveDisplay.from_predictions(
      y_test_cd.iloc[:, 1]
      , y_pred[:, 1],
      name=f"priority vs the rest",
      color="darkorange",
      plot_chance_level=True, )
```

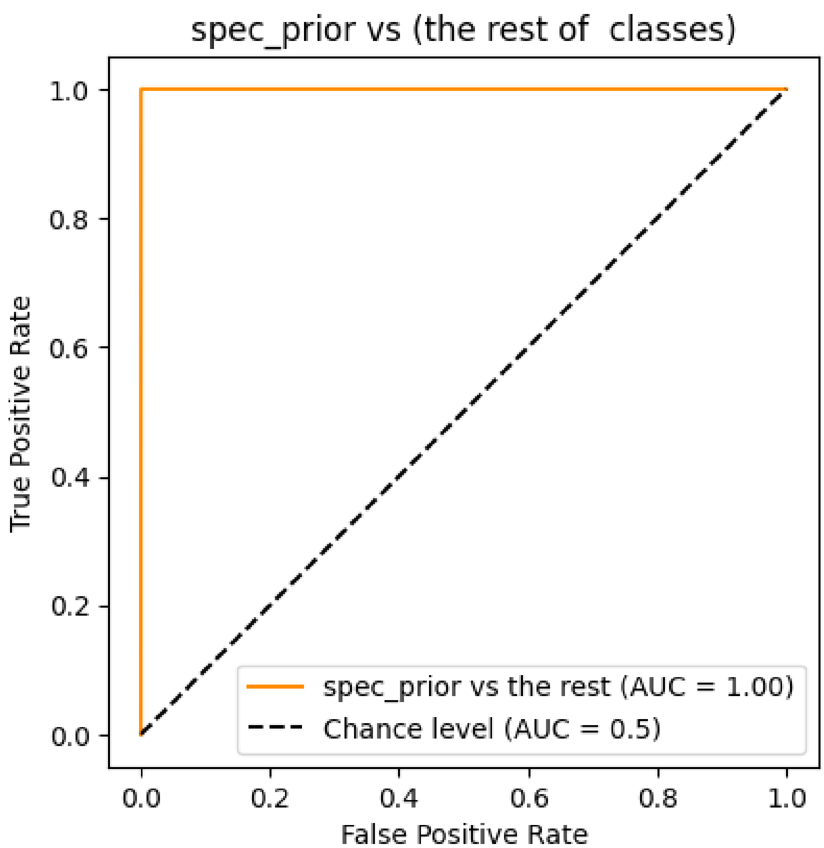
```
plt.axis("square")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("priority vs (the rest of  classes)")
plt.legend()
plt.show()
```



[79]:
```
# plot de courbe ROC pour la classe 'recommend'
RocCurveDisplay.from_predictions(
      y_test_cd.iloc[:, 2]
      , y_pred[:, 2],
      name=f"recommend vs the rest",
      color="darkorange",
      plot_chance_level=True, )
plt.axis("square")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("recommend vs (the rest of  classes)")
plt.legend()
plt.show()
```



[80]:
```
# plot de courbe ROC pour la classe 'spec_prior'
RocCurveDisplay.from_predictions(
      y_test_cd.iloc[:, 2]
      , y_pred[:, 2],
      name=f"spec_prior vs the rest",
      color="darkorange",
      plot_chance_level=True, )
plt.axis("square")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.title("spec_prior vs (the rest of  classes)")
plt.legend()
plt.show()
```



0.0.7 Les arbres de decision

[81]:
```
#equilbrer les classes
sample_weights = compute_sample_weight('balanced', y_train)
```

[82]:
```
#Entrainement du modele
"""
class_weight='balanced' indique que l'arbre tiendra compte du déséquilibre de␣
 ↪classe lors de la construction de l'arbre.
"""
Arbr = DecisionTreeClassifier(criterion='gini', class_weight='balanced',␣
 ↪random_state=random_seed)
Arbr.fit(X_train, y_train)
```

[82]:
```
DecisionTreeClassifier(class_weight='balanced', random_state=42)
```

[83]:
```
#On effectue la tache du test
y_pred = Arbr.predict(X_test_cd)
```

[84]:
```
#Bilan d'evaluation du perfomance du modele :

print(classification_report(y_test_cd, y_pred))
print(accuracy_score(y_test_cd, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1320
           1       0.99      0.99      0.99      1272
           2       1.00      1.00      1.00       106
           3       0.99      0.99      0.99      1190

   micro avg       0.99      0.99      0.99      3888
   macro avg       0.99      0.99      0.99      3888
weighted avg       0.99      0.99      0.99      3888
 samples avg       0.99      0.99      0.99      3888

0.9933127572016461
```

0.0.8 MLP

[85]:
```
# construction de l'architecture a l'aide de l'API TensorFlow
model = models.Sequential([
    layers.Dense(64, activation='relu', input_shape =(X_train.shape[1],)),
    layers.Dense(32, activation = 'relu'),
    layers.Dense(4, activation ='softmax')
])
model.compile(optimizer = 'adam', loss = 'categorical_crossentropy', metrics =
 ↪['accuracy'])
```

[86]:
```
#tester s'il ya un problem au niveau X_train
X_train.shape
```

[86]: (9072, 27)

[87]:
```
# Entraienemt du modele
model.fit(X_train, y_train , epochs = 40, batch_size = 32, validation_split = 0.
 ↪2)
```

```
Epoch 1/40
227/227 [==============================] - 2s 4ms/step - loss: 0.6176 -
accuracy: 0.7930 - val_loss: 0.2336 - val_accuracy: 0.9229
Epoch 2/40
227/227 [==============================] - 1s 3ms/step - loss: 0.1955 -
accuracy: 0.9260 - val_loss: 0.1612 - val_accuracy: 0.9438
Epoch 3/40
227/227 [==============================] - 1s 3ms/step - loss: 0.1330 -
accuracy: 0.9514 - val_loss: 0.1202 - val_accuracy: 0.9521
```

```
Epoch 4/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0955 -
accuracy: 0.9671 - val_loss: 0.0935 - val_accuracy: 0.9664
Epoch 5/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0682 -
accuracy: 0.9797 - val_loss: 0.0623 - val_accuracy: 0.9824
Epoch 6/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0491 -
accuracy: 0.9890 - val_loss: 0.0451 - val_accuracy: 0.9895
Epoch 7/40
227/227 [==============================] - 1s 2ms/step - loss: 0.0356 -
accuracy: 0.9931 - val_loss: 0.0352 - val_accuracy: 0.9890
Epoch 8/40
227/227 [==============================] - 1s 2ms/step - loss: 0.0268 -
accuracy: 0.9952 - val_loss: 0.0254 - val_accuracy: 0.9939
Epoch 9/40
227/227 [==============================] - 1s 2ms/step - loss: 0.0192 -
accuracy: 0.9972 - val_loss: 0.0187 - val_accuracy: 0.9978
Epoch 10/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0136 -
accuracy: 0.9994 - val_loss: 0.0165 - val_accuracy: 0.9978
Epoch 11/40
227/227 [==============================] - 1s 2ms/step - loss: 0.0100 -
accuracy: 0.9997 - val_loss: 0.0111 - val_accuracy: 0.9989
Epoch 12/40
227/227 [==============================] - 1s 2ms/step - loss: 0.0073 -
accuracy: 1.0000 - val_loss: 0.0094 - val_accuracy: 0.9978
Epoch 13/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0056 -
accuracy: 0.9999 - val_loss: 0.0072 - val_accuracy: 0.9989
Epoch 14/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0045 -
accuracy: 0.9999 - val_loss: 0.0052 - val_accuracy: 0.9994
Epoch 15/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0032 -
accuracy: 1.0000 - val_loss: 0.0047 - val_accuracy: 0.9994
Epoch 16/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0026 -
accuracy: 1.0000 - val_loss: 0.0037 - val_accuracy: 1.0000
Epoch 17/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0020 -
accuracy: 1.0000 - val_loss: 0.0030 - val_accuracy: 1.0000
Epoch 18/40
227/227 [==============================] - 1s 3ms/step - loss: 0.0016 -
accuracy: 1.0000 - val_loss: 0.0027 - val_accuracy: 1.0000
Epoch 19/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0013 -
accuracy: 1.0000 - val_loss: 0.0024 - val_accuracy: 1.0000
```

```
Epoch 20/40
227/227 [==============================] - 1s 4ms/step - loss: 0.0011 -
accuracy: 1.0000 - val_loss: 0.0022 - val_accuracy: 0.9994
Epoch 21/40
227/227 [==============================] - 1s 3ms/step - loss: 9.3037e-04 -
accuracy: 1.0000 - val_loss: 0.0016 - val_accuracy: 1.0000
Epoch 22/40
227/227 [==============================] - 1s 2ms/step - loss: 7.3420e-04 -
accuracy: 1.0000 - val_loss: 0.0017 - val_accuracy: 1.0000
Epoch 23/40
227/227 [==============================] - 1s 3ms/step - loss: 6.1836e-04 -
accuracy: 1.0000 - val_loss: 0.0015 - val_accuracy: 1.0000
Epoch 24/40
227/227 [==============================] - 1s 2ms/step - loss: 5.2987e-04 -
accuracy: 1.0000 - val_loss: 0.0014 - val_accuracy: 0.9994
Epoch 25/40
227/227 [==============================] - 1s 3ms/step - loss: 4.4069e-04 -
accuracy: 1.0000 - val_loss: 0.0010 - val_accuracy: 1.0000
Epoch 26/40
227/227 [==============================] - 1s 3ms/step - loss: 3.6816e-04 -
accuracy: 1.0000 - val_loss: 0.0011 - val_accuracy: 1.0000
Epoch 27/40
227/227 [==============================] - 1s 2ms/step - loss: 3.0817e-04 -
accuracy: 1.0000 - val_loss: 9.4806e-04 - val_accuracy: 1.0000
Epoch 28/40
227/227 [==============================] - 1s 2ms/step - loss: 2.7032e-04 -
accuracy: 1.0000 - val_loss: 7.9171e-04 - val_accuracy: 1.0000
Epoch 29/40
227/227 [==============================] - 1s 2ms/step - loss: 2.3931e-04 -
accuracy: 1.0000 - val_loss: 7.6222e-04 - val_accuracy: 1.0000
Epoch 30/40
227/227 [==============================] - 1s 2ms/step - loss: 1.9482e-04 -
accuracy: 1.0000 - val_loss: 6.9287e-04 - val_accuracy: 1.0000
Epoch 31/40
227/227 [==============================] - 1s 2ms/step - loss: 1.7157e-04 -
accuracy: 1.0000 - val_loss: 6.1707e-04 - val_accuracy: 1.0000
Epoch 32/40
227/227 [==============================] - 1s 3ms/step - loss: 1.4294e-04 -
accuracy: 1.0000 - val_loss: 7.5702e-04 - val_accuracy: 1.0000
Epoch 33/40
227/227 [==============================] - 1s 2ms/step - loss: 1.2965e-04 -
accuracy: 1.0000 - val_loss: 5.0513e-04 - val_accuracy: 1.0000
Epoch 34/40
227/227 [==============================] - 1s 2ms/step - loss: 1.0811e-04 -
accuracy: 1.0000 - val_loss: 4.1172e-04 - val_accuracy: 1.0000
Epoch 35/40
227/227 [==============================] - 1s 2ms/step - loss: 9.1927e-05 -
accuracy: 1.0000 - val_loss: 3.8087e-04 - val_accuracy: 1.0000
```

```
Epoch 36/40
227/227 [==============================] - 1s 3ms/step - loss: 7.8933e-05 -
accuracy: 1.0000 - val_loss: 3.8118e-04 - val_accuracy: 1.0000
Epoch 37/40
227/227 [==============================] - 1s 3ms/step - loss: 6.9447e-05 -
accuracy: 1.0000 - val_loss: 3.7739e-04 - val_accuracy: 1.0000
Epoch 38/40
227/227 [==============================] - 1s 4ms/step - loss: 5.9119e-05 -
accuracy: 1.0000 - val_loss: 3.1292e-04 - val_accuracy: 1.0000
Epoch 39/40
227/227 [==============================] - 1s 4ms/step - loss: 5.1588e-05 -
accuracy: 1.0000 - val_loss: 2.4858e-04 - val_accuracy: 1.0000
Epoch 40/40
227/227 [==============================] - 1s 3ms/step - loss: 4.4205e-05 -
accuracy: 1.0000 - val_loss: 2.5261e-04 - val_accuracy: 1.0000
```

[87]:
```
<keras.src.callbacks.History at 0x7f1ac00e1750>
```

[88]:
```python
# On convertit y_true en Array
y_true = y_test_cd.values
y_true
```

[88]:
```
array([[1, 0, 0, 0],
       [0, 0, 0, 1],
       [0, 1, 0, 0],
       …,
       [1, 0, 0, 0],
       [0, 0, 0, 1],
       [0, 0, 0, 1]], dtype=uint8)
```

[89]:
```python
# Extraire les étiquettes de classe en utilisant argmax
y_true_classes = np.argmax(y_true, axis=1)
y_true_classes
```

[89]:
```
array([0, 3, 1, …, 0, 3, 3])
```

[90]:
```python
# Extraire les étiquettes de classe en utilisant argmax
y_pred_classes = np.argmax(y_pred, axis=1)
y_pred_classes.size
```

[90]:
```
3888
```

[91]:
```python
# On costruit MLP_accuracy qui sert a calculer l'accuracy du modele :
def MLP_accuracy(y_pred_classes,y_true_classes) :
  k=0
  for i in range(y_pred_classes.size):
     if  y_pred_classes[i] == y_true_classes[i] :
```

```
        k += 1
    print("Accuracy :",(k/y_pred_classes.size))\

MLP_accuracy(y_pred_classes,y_true_classes)
```

```
Accuracy : 0.9933127572016461
```

[92]:

```
# construire  la matrice de confusion
conf_matrix = confusion_matrix(y_true.argmax(axis=1), y_pred_classes)

# We plot confusion matrix using seaborn heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt="d", cmap="Blues", xticklabels=np.
 ↪unique(y_true_classes), yticklabels=np.unique(y_true_classes))
plt.title('Confusion Matrix for Multiclass Classification')
plt.xlabel('Predicted')
plt.ylabel('True')
plt.show()
```