Importem les llibreries necessaries per a la realització del projecte.

```
import pandas as pd
from pandas import plotting
import numpy as np
from sklearn.preprocessing import LabelEncoder
import matplotlib.pyplot as plt
import seaborn as sns
```

1. Preprocessament de Dades

En aquesta secció, realitzarem el pre-processament de les dades per preparar-les per a la construcció de models de detecció de phishing.

1.1 Lectura de Dades

Primer, llegim el conjunt de dades des del fitxer CSV i mostrem les primeres files per entendre la seva estructura.

```
data = pd.read_csv("/dataset_phishing.csv")
data.head()
```

	url	length_url	length_hostname	ip	nb_dots
0	http://www.crestonwood.com/router.php	37	19	0	3
1	http://shadetreetechnology.com/V4/validation/a...	77	23	1	1
2	https://support-appleId.com.secureupdate.duila...	126	50	1	4
3	http://rgipt.ac.in	18	11	0	2
4	http://www.iracing.com/tracks/gateway-motorspo...	55	15	0	2

5 rows × 89 columns

1.2 Anàlisi Exploratori de Dades

A continuació, realitzem un anàlisi exploratori de les dades per comprendre millor el conjunt de dades. Això inclou una descripció estadística, la comprovació de valors nuls, la identificació dels tipus de dades, la informació sobre el nombre de files i columnes, i la cerca de files duplicades.

```
# Obtenim una descripció completa del dataset.
data.describe([x*0.1 for x in range(10)])
```

	length_url	length_hostname	ip	nb_dots	nb_hyphens

```
# Informació sobre el nombre de files i columnes.
data.info()
```

```
 33  nb_subdomains               11430 non-null  int64
 34  prefix_suffix               11430 non-null  int64
 35  random_domain               11430 non-null  int64
 36  shortening_service          11430 non-null  int64
 37  path_extension              11430 non-null  int64
 38  nb_redirection              11430 non-null  int64
 39  nb_external_redirection     11430 non-null  int64
 40  length_words_raw            11430 non-null  int64
 41  char_repeat                 11430 non-null  int64
 42  shortest_words_raw          11430 non-null  int64
 43  shortest_word_host          11430 non-null  int64
 44  shortest_word_path          11430 non-null  int64
 45  longest_words_raw           11430 non-null  int64
 46  longest_word_host           11430 non-null  int64
 47  longest_word_path           11430 non-null  int64
 48  avg_words_raw               11430 non-null  float64
 49  avg_word_host               11430 non-null  float64
 50  avg_word_path               11430 non-null  float64
 51  phish_hints                 11430 non-null  int64
 52  domain_in_brand             11430 non-null  int64
 53  brand_in_subdomain          11430 non-null  int64
 54  brand_in_path               11430 non-null  int64
 55  suspecious_tld              11430 non-null  int64
 56  statistical_report          11430 non-null  int64
 57  nb_hyperlinks               11430 non-null  int64
 58  ratio_intHyperlinks         11430 non-null  float64
 59  ratio_extHyperlinks         11430 non-null  float64
 60  ratio_nullHyperlinks        11430 non-null  int64
 61  nb_extCSS                   11430 non-null  int64
 62  ratio_intRedirection        11430 non-null  int64
 63  ratio_extRedirection        11430 non-null  float64
 64  ratio_intErrors             11430 non-null  int64
 65  ratio_extErrors             11430 non-null  float64
 66  login_form                  11430 non-null  int64
 67  external_favicon            11430 non-null  int64
 68  links_in_tags               11430 non-null  float64
 69  submit_email                11430 non-null  int64
 70  ratio_intMedia              11430 non-null  float64
 71  ratio_extMedia              11430 non-null  float64
 72  sfh                         11430 non-null  int64
 73  iframe                      11430 non-null  int64
 74  popup_window                11430 non-null  int64
 75  safe_anchor                 11430 non-null  float64
 76  onmouseover                 11430 non-null  int64
 77  right_clic                  11430 non-null  int64
 78  empty_title                 11430 non-null  int64
 79  domain_in_title             11430 non-null  int64
 80  domain_with_copyright       11430 non-null  int64
 81  whois_registered_domain     11430 non-null  int64
 82  domain_registration_length  11430 non-null  int64
 83  domain_age                  11430 non-null  int64
 84  web_traffic                 11430 non-null  int64
 85  dns_record                  11430 non-null  int64
 86  google_index                11430 non-null  int64
 87  page_rank                   11430 non-null  int64
 88  status                      11430 non-null  object
dtypes: float64(13), int64(74), object(2)
memory usage: 7.8+ MB
```

```
# Nombre de valors nuls en cada columna.
data.isnull().sum()
```

```
url                0
length_url         0
length_hostname    0
ip                 0
nb_dots            0
                  ..
web_traffic        0
dns_record         0
google_index       0
page_rank          0
status             0
Length: 89, dtype: int64
```

```
# Nombre de files duplicades.
data.duplicated().sum()
```

```
0
```

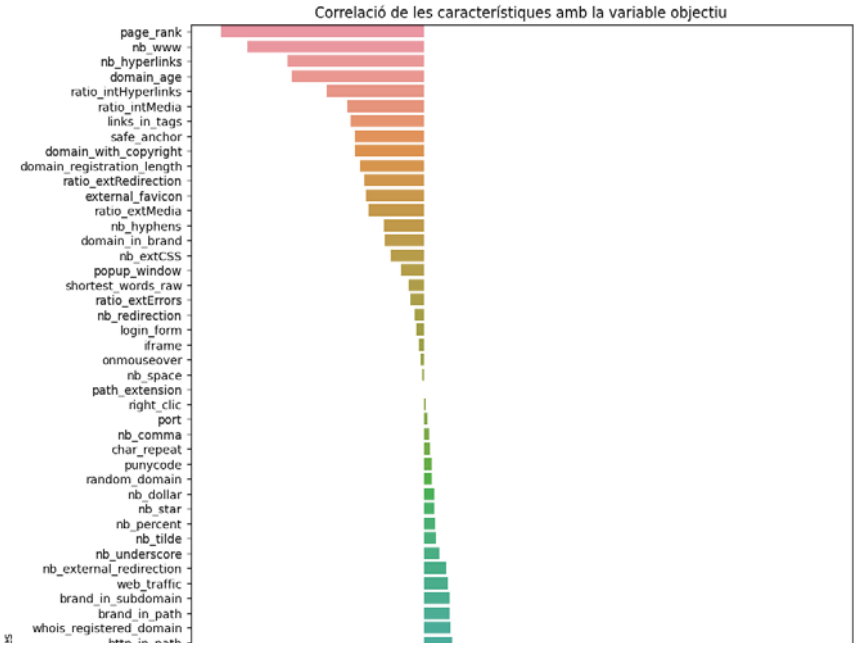
1.2.1 Correlació de caracteristiques

Avaluarem la correlació de cada característica amb la variable objectiu per obtenir un sentit inicial de la seva importància. Les característiques que tenen una correlació molt baixa es poden considerar per a l'eliminació.

```
# Convertim la columna 'status' a valors numèrics (1 per a 'phishing' i 0 per a 'legitimate')
data['status_numeric'] = data['status'].apply(lambda x: 1 if x == 'phishing' else 0)

# Calculem la correlació de cada característica amb la columna 'status_numeric'
correlations = data.drop(columns=['url', 'status']).corrwith(data['status_numeric']).sort_values()

# Visualitzem les correlacions
plt.figure(figsize=(10, 20))
sns.barplot(x=correlations, y=correlations.index)
plt.title('Correlació de les característiques amb la variable objectiu')
plt.xlabel('Correlació')
plt.ylabel('Característiques')
plt.show()
```



A partir del diagrama de correlació, podem veure com cada característica es correlaciona amb la variable de destinació.

Important:

- Una correlació propera a 1 indica una forta relació positiva.
- Una correlació propera a -1 indica una forta relació negativa.
- Una correlació propera a 0 indica poca o cap relació.

Basant-nos en això, podríem considerar eliminar característiques que tenen una correlació molt propera a 0, ja que podrien no contribuir molt al poder predictiu d'un model. No obstant això, els mantindrem de moment i podríem revisar aquesta decisió després de l'avaluació del model.

1.3 Manipulació de Dades

En aquesta secció, realitzem les següents etapes de pre-processament de dades:

En primer lloc, haurem de guardar en una nova variable el conjunt de dades original per a poder fer modificacions sobre aquesta.

```
data_preprocessada = data
```

Eliminarem la columna "url" ja que és única per a cada fila i no aportarà informació útil per al model de classificació. En aquest cas, estem simplificant el conjunt de dades eliminant una característica que no és rellevant per a la nostra anàlisi.

```
data_preprocessada = data_preprocessada.drop(columns=['url'])
data_preprocessada.head()
```

	length_url	length_hostname	ip	nb_dots	nb_hyphens	nb_at	nb_qm	nb_and	nb_or
0	37	19	0	3	0	0	0	0	0
1	77	23	1	1	0	0	0	0	0
2	126	50	1	4	1	0	1	2	0
3	18	11	0	2	0	0	0	0	0
4	55	15	0	2	2	0	0	0	0

5 rows × 89 columns

1.3.1 Tractament de Valors Nuls

· Per a les **columnes numèriques**, substituïm els valors nuls per la mitjana de la columna.
· Per a les **columnes categòriques**, substituïm els valors nuls pel valor més freqüent de la columna.

```
# Tractament de valors nuls
for column in data_preprocessada.select_dtypes(include=['int', 'float']).columns:
    data_preprocessada[column].fillna(data_preprocessada[column].mean(), inplace=True)

for column in data_preprocessada.select_dtypes(include=['object']).columns:
    data_preprocessada[column].fillna(data_preprocessada[column].mode()[0], inplace=True)
```

1.3.2 Eliminar columnes Redundants

Simplificarem el conjunt de dades. Analitzem si hi ha alguna columna que pugui ser redundant o no útil per a la nostra anàlisi. Revisem les estadístiques descriptives de les columnes per veure si alguna d'elles presenta variància zero o valors constants, ja que aquestes columnes no aportarien informació útil per a la majoria d'algorismes.

```
# Obtenim les estadistiques de les columnes numeriques.
desc_stats = data_preprocessada.describe()

# Identifiquem les columnes amb 0 variança (desviació estàndard igual a zero)
zero_variance_columns = desc_stats.columns[desc_stats.loc["std"] == 0]

zero_variance_columns
```

```
Index(['nb_or', 'ratio_nullHyperlinks', 'ratio_intRedirection',
       'ratio_intErrors', 'submit_email', 'sfh'],
      dtype='object')
```

Les columnes següents presenten variància zero, la qual cosa significa que tenen el mateix valor en totes les files:

- nb_or
- ratio_nullHyperlinks
- ratio_intRedirection
- ratio_intErrors
- submit_email
- sfh

Aquestes columnes poden ser considerades com a redundants ja que no aporten informació variable al nostre model. Per tant, es recomana eliminar-les del conjunt de dades.

A continuació, procedirem a eliminar aquestes columnes del conjunt de dades.

```
# Remove the zero variance columns from the dataset
data_preprocessada = data_preprocessada.drop(columns=zero_variance_columns)

# Display the first few rows of the cleaned dataset
data_preprocessada.head()
```

	length_url	length_hostname	ip	nb_dots	nb_hyphens	nb_at	nb_qm	nb_and	nb_eq
0	37	19	0	3	0	0	0	0	0
1	77	23	1	1	0	0	0	0	0
2	126	50	1	4	1	0	1	2	3
3	18	11	0	2	0	0	0	0	0
4	55	15	0	2	2	0	0	0	0

5 rows × 83 columns

1.3.3 Conversió de Dades Categòriques a Dades Numèriques

· Utilitzem **LabelEncoder** per transformar les columnes categòriques en dades numèriques.

```
# Conversió de dades categòriques a numèriques
le = LabelEncoder()
for column in data_preprocessada.columns:
    if data_preprocessada[column].dtype == type(object):
        data_preprocessada[column] = le.fit_transform(data_preprocessada[column])
```

1.3.4 Normalització de Dades Numèriques

Normalitzem les dades numèriques en un rang de 0 a 1.

```
# Normalització de dades numèriques
for column in data_preprocessada.columns:
    if data_preprocessada[column].dtype in ['int16', 'int32', 'int64', 'float16', 'float32', 'float64']:
        data_preprocessada[column] = (data_preprocessada[column] - data_preprocessada[column].min()) / (data_preprocessada[column].max()
```

Ara que les dades han estat pre-processades, estan llestes per ser utilitzades per a la construcció i avaluació de models de detecció de phishing.

```
# Dades pre-processades
data_preprocessada.head()
```

	length_url	length_hostname	ip	nb_dots	nb_hyphens	nb_at	nb_qm	nb_and	
0	0.015347	0.071429	0.0	0.086957	0.000000	0.0	0.000000	0.000000	C
1	0.039902	0.090476	1.0	0.000000	0.000000	0.0	0.000000	0.000000	C
2	0.069982	0.219048	1.0	0.130435	0.023256	0.0	0.333333	0.105263	C
3	0.003683	0.033333	0.0	0.043478	0.000000	0.0	0.000000	0.000000	C
4	0.026397	0.052381	0.0	0.043478	0.046512	0.0	0.000000	0.000000	C

5 rows × 83 columns

1.4 Histogrames i Boxplot

Crearem gràfics de caixes (boxplots) i histogrames per visualitzar la distribució de les característiques numèriques en el conjunt de dades pre-processades i originals. Aquests gràfics ens ajudaran a entendre millor com es distribueixen les dades i a identificar valors extrems (outliers).

1.4.1 Gràfics de Caixes (Boxplots)

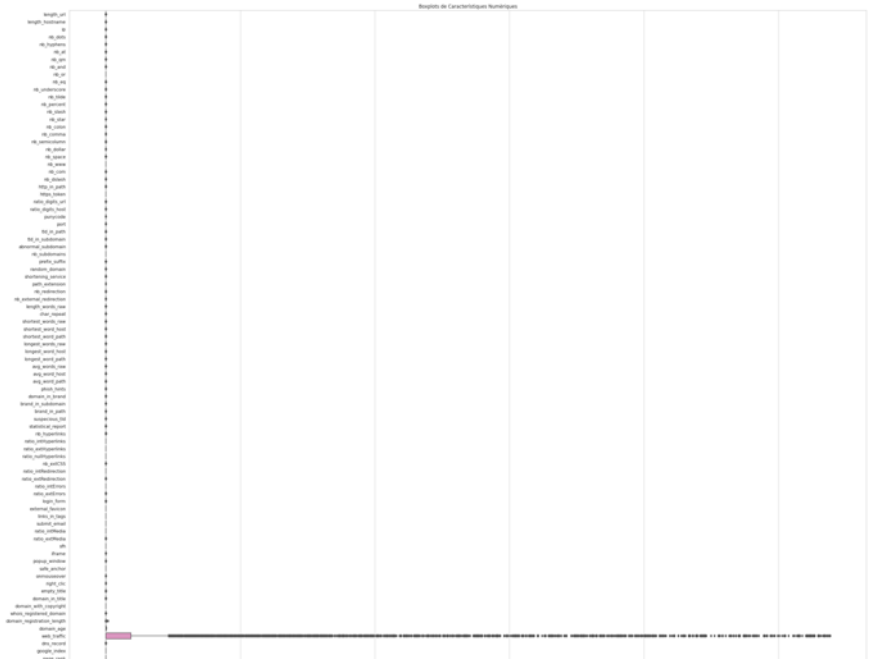
Els gràfics de caixes són útils per representar visualment la distribució i la variabilitat de les dades numèriques. Ens mostren la mediana, els quartils i els valors extrems

· Dades originals

```
# Creació de gràfics de caixes més grans per a característiques numèriques
plt.figure(figsize=(36, 30))
sns.set(style="whitegrid")

# Seleccionem les característiques numèriques per als quals volem generar els boxplots
features_for_boxplot = data.select_dtypes(include=['int', 'float'])

# Dibuixem els gràfics de caixes
sns.boxplot(data=features_for_boxplot, orient="h", palette="Set2")
plt.title("Boxplots de Característiques Numèriques")
plt.xlabel("Valor")
plt.show()
```

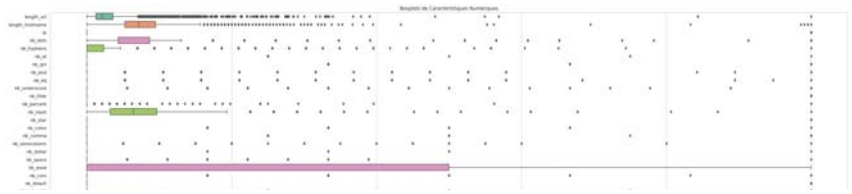


· **Dades pre-processades**

```
# Creació de gràfics de caixes més grans per a característiques numèriques
plt.figure(figsize=(36, 30))
sns.set(style="whitegrid")

# Seleccionem les característiques numèriques per als quals volem generar els boxplots
features_for_boxplot = data_preprocessada.select_dtypes(include=['int', 'float'])

# Dibuixem els gràfics de caixes
sns.boxplot(data=features_for_boxplot, orient="h", palette="Set2")
plt.title("Boxplots de Característiques Numèriques")
plt.xlabel("Valor")
plt.show()
```



1.4.1 Histogrames

Els histogrames mostren la distribució de les dades numèriques i ens permeten veure la forma de la distribució.

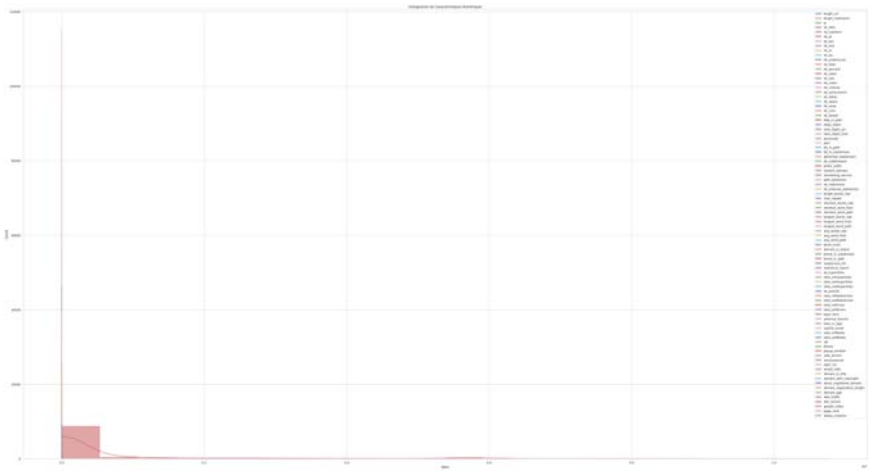
· Dades originals

```
# Creació d'histogrames més grans per a característiques numèriques
plt.figure(figsize=(56, 30))
sns.set(style="whitegrid")

# Seleccionem les característiques numèriques per als quals volem generar histogrames
features_for_histogram = data.select_dtypes(include=['int', 'float'])

# Dibuixem els histogrames
for feature in features_for_histogram.columns:
    sns.histplot(data[feature], kde=True, bins=20, label=feature)

plt.title("Histogrames de Característiques Numèriques")
plt.xlabel("Valor")
plt.legend()
plt.show()
```



· Dades pre-processades

```
# Creació d'histogrames més grans per a característiques numèriques
plt.figure(figsize=(56, 30))
sns.set(style="whitegrid")

# Seleccionem les característiques numèriques per als quals volem generar histogrames
features_for_histogram = data_preprocessada.select_dtypes(include=['int', 'float'])

# Dibuixem els histogrames
for feature in features_for_histogram.columns:
    sns.histplot(data_preprocessada[feature], kde=True, bins=20, label=feature)

plt.title("Histogrames de Característiques Numèriques")
plt.xlabel("Valor")
```

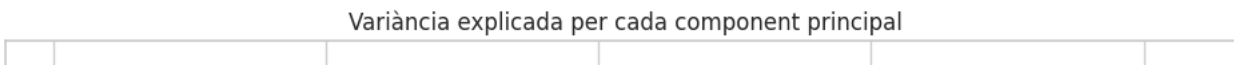
```
plt.legend()
plt.show()
```

1.5 PCA (Principal Component Analysis)

```
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt

# Apliquem PCA
pca = PCA()
components = pca.fit_transform(data_preprocessada)

# Visualitzem la variància explicada per cada component principal
plt.figure(figsize=(12, 6))
plt.bar(range(1, len(pca.explained_variance_ratio_) + 1), pca.explained_variance_ratio_)
plt.xlabel('Component principal')
plt.ylabel('Variància explicada')
plt.title('Variància explicada per cada component principal')
plt.show()
```



A partir de la visualització, podem fer les següents observacions:

1. Declivi ràpid inicial: Les primeres components principals expliquen una quantitat significativa de la variabilitat de les dades, amb un declivi

2. Plateau després d'unes poques components: Després d'un cert número de components (aproximadament 10-15), la variància explicada per cada compor

Aquestes observacions suggerixen que podríem reduir la dimensió del conjunt de dades utilitzant només les primeres components principals sense perdre massa informació. Aquesta reducció podria ser útil per visualitzar les dades o per accelerar l'entrenament de models de machine learning.

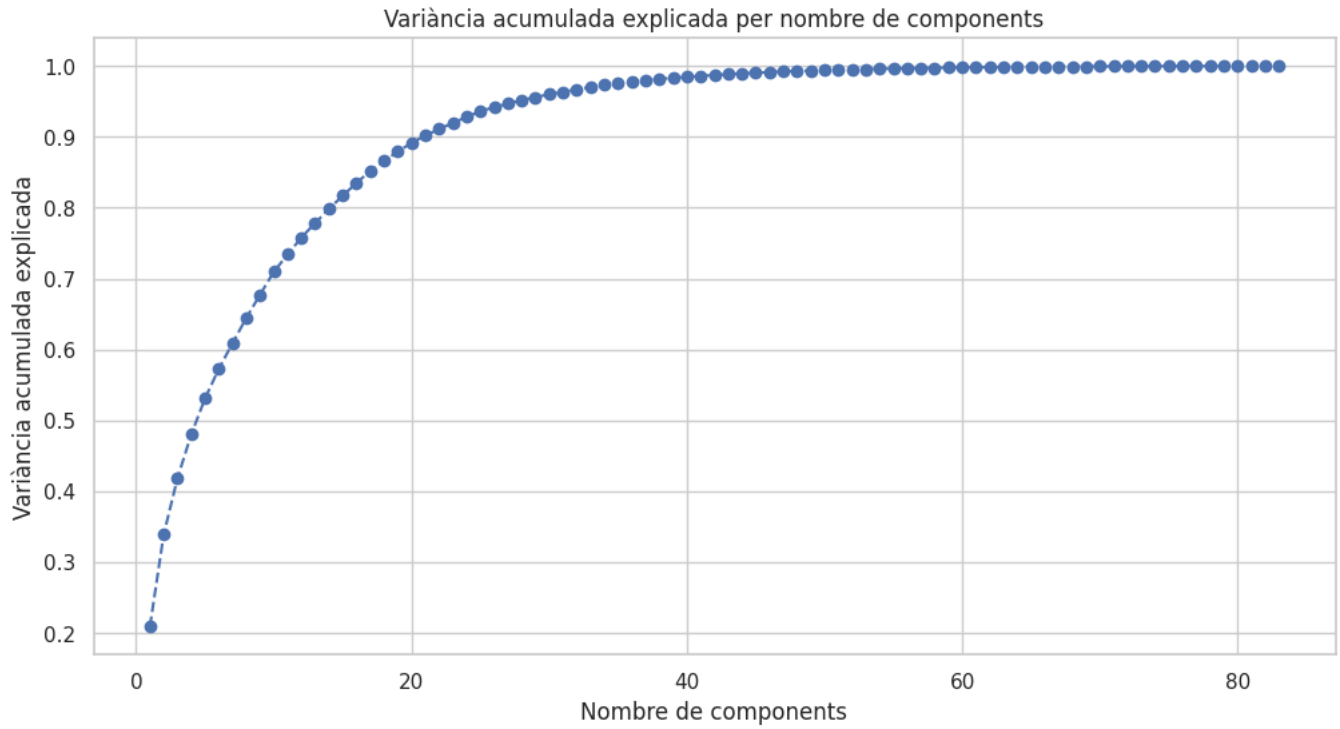
A continuació, podem calcular la variància acumulada explicada per les components per determinar quantes d'elles s'haurien d'utilitzar per capturar una determinada quantitat de la variabilitat total (per exemple, el 95%).

```
# Calculem la variància acumulada explicada
variancia_acumulada = pca.explained_variance_ratio_.cumsum()

# Visualitzem la variància acumulada
plt.figure(figsize=(12, 6))
plt.plot(range(1, len(variancia_acumulada) + 1), variancia_acumulada, marker='o', linestyle='--')
plt.xlabel('Nombre de components')
plt.ylabel('Variància acumulada explicada')
plt.title('Variància acumulada explicada per nombre de components')
plt.show()

# Determinem el nombre de components necessaris per explicar almenys el 95% de la variància
components_95_variancia = sum(variancia_acumulada < 0.95) + 1

components_95_variancia
```



28

La visualització mostra la variància acumulada explicada a mesura que augmentem el nombre de components principals. A partir d'aquesta, podem observar que:

1. La variància acumulada augmenta ràpidament amb les primeres components i comença a estabilitzar-se a mesura que s'afegeixen més components.

2. Per explicar almenys el 95% de la variància total de les dades, necessitaríem utilitzar 29 components principals.

```
# Reajustem el PCA amb 29 components
pca_29 = PCA(n_components=components_95_variancia)
data_reduida = pca_29.fit_transform(data_preprocessada)
```

```
# Mostrem les primeres files de la representació reduïda
data_reduida[:5]
```

```
array([[-5.87753474e-01, -3.92852283e-01, -7.47331852e-02,
        -3.98683357e-01,  1.00149249e-01, -3.73004788e-01,
         5.55729505e-01, -1.73783746e-01,  5.45759359e-01,
         9.49601250e-01,  4.17432245e-01, -1.71030247e-01,
        -1.68459537e-01,  1.72859231e-01, -4.07327029e-01,
        -1.28559058e-01,  5.13687747e-02, -3.28358121e-01,
         1.31309860e-01,  2.80069822e-01,  1.69721609e-01,
         6.42376608e-02,  2.09590995e-01,  4.15183857e-01,
        -2.92346493e-01,  3.02826150e-01,  9.64697135e-02,
         3.89648555e-02],
       [ 6.85830601e-01, -1.13109141e+00,  7.12679707e-02,
         1.53777289e-02, -6.16453448e-01,  1.93272060e-01,
         3.16311812e-01,  3.94485664e-01, -1.36343341e-01,
        -4.91845191e-01, -5.97256706e-01,  3.26679638e-01,
        -5.89441169e-01, -9.17512986e-02, -1.28193338e-01,
         6.59584878e-02,  1.86485694e-01,  1.01376982e-01,
        -1.02276249e-01,  1.30442258e-01, -3.29928616e-01,
        -6.47693121e-02, -1.66631293e-01,  8.60760558e-02,
         6.96455697e-02, -1.96110445e-02, -2.76470449e-02,
        -2.01168005e-01],
       [ 9.70201563e-01, -1.18075737e+00,  2.60307069e-01,
         8.24105722e-01,  2.03805921e-01,  1.23293748e+00,
        -9.58494007e-02, -3.28720679e-01, -2.72912646e-01,
        -3.75061121e-01, -1.56066273e-01, -1.47876839e-02,
        -2.32361055e-02,  8.45827419e-02,  3.05216614e-01,
        -8.25070622e-02,  7.67176803e-04, -2.60104093e-01,
         1.89563264e-01,  5.79088480e-02,  5.65898152e-01,
        -6.24307036e-03, -7.76616467e-02, -1.94133127e-01,
         1.86889114e-01,  3.16613935e-02, -1.92884280e-01,
         9.93273822e-02],
       [-8.54530855e-01, -7.67436012e-01, -5.58668640e-01,
        -3.88911142e-01, -3.81803326e-01, -1.26887992e-01,
        -1.06891254e-01,  7.84168196e-02, -1.57251313e-01,
        -4.55211956e-01,  5.53620491e-02,  2.47427282e-01,
        -5.05103488e-02, -5.87858341e-02, -2.06109369e-01,
         8.01607004e-03, -1.12426115e-01, -8.54927682e-02,
        -2.50851442e-02,  1.95654976e-02,  1.01300879e-01,
         1.09284434e-01, -4.71612033e-02,  4.97287014e-02,
        -3.31509830e-01,  7.59082315e-02,  3.01922157e-02,
         2.54806823e-01],
       [-9.85874333e-01,  5.77977742e-01, -2.33560880e-01,
        -5.05703614e-01,  2.48440085e-01,  5.39213899e-01,
         4.58031031e-01,  5.48733352e-01,  4.99310313e-01,
         6.37855008e-01,  4.53986498e-01,  3.63066417e-01,
        -2.56153701e-01, -3.05464684e-01, -1.22999167e-01,
         3.68043634e-02,  6.77994716e-02,  3.29300373e-01,
         7.01751158e-01, -1.65186312e-01, -3.61886123e-02,
        -2.15982333e-01, -1.56897433e-01,  6.16923422e-02,
         2.62704039e-01, -3.35504296e-02, -4.87407968e-02,
        -5.87800451e-03]])
```