Trabajo Semana 2 -Análisis exploratorio de datos

Asignatura: Machine Learning ¶

Especialización en Inteligencia Artificial

Realizado por: Michael Andrés Mora Poveda

Hola a todos, el objectivo de este trabajo es realizar el análisis exploratorio de datos para el dataset Congressional Voting Records Data Set el cual se puede encontrar en la siguiente página web (y dentro de este folder también)y el cual se encuentra referenciado al final de este notebook:

- https://archive.ics.uci.edu/ml/datasets/congressional+voting+records (https://archive.ics.uci.edu/ml/datasets/congressional+voting+records)

Además, es importante tener en cuenta que dentro de este análisis se aplicarán varias librerías de Python para mayor practicidad.

Las siguientes descripciones serán útiles para conocer la estructura de los datasets:

Descripción general:

Según el repositorio oficial, el dataset contiene los votos del **Congressional Quarterly Almanac - Edition 98th** de los Estados Unidos por partidos o casas políticas del año 1984 en donde se describen 9 tipos de voto:

1. voted for
2. paired for
3. announced for
4. voted against
5. paired against
6. announced against
7. voted present
8. voted present to avoid conflict of interest
9. did not vote

Nota: Los 3 primeros bullets simplifican el voto para aceptar la propuesta, el segundo grupo de 3 bullets simplifican el voto para no aceptar la propuesta y el último grupo de 3 bullets para no votar o evitar conflicto de intereses.

Atributos (columnas o features)

De acuerdo al repositorio referenciado, los siguientes temas fueron discutidos y puestos a votación junto a el tipo de decisión tomada (yes/no decision):

1. Class Name: 2 (democrat, republican)
2. handicapped-infants: 2 (y,n)
3. water-project-cost-sharing: 2 (y,n)
4. adoption-of-the-budget-resolution: 2 (y,n)
5. physician-fee-freeze: 2 (y,n)
6. el-salvador-aid: 2 (y,n)
7. religious-groups-in-schools: 2 (y,n)
8. anti-satellite-test-ban: 2 (y,n)
9. aid-to-nicaraguan-contras: 2 (y,n)
10. mx-missile: 2 (y,n)
11. immigration: 2 (y,n)
12. synfuels-corporation-cutback: 2 (y,n)
13. education-spending: 2 (y,n)
14. superfund-right-to-sue: 2 (y,n)
15. crime: 2 (y,n)
16. duty-free-exports: 2 (y,n)
17. export-administration-act-south-africa: 2 (y,n)

1. Análisis exploratorio de datos dataset Congress voting records 1984

Vamos a revisar la estructura general de los datos:

In [1]:
```
#Se importan las librerías clásicas:
import seaborn as sns
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
```

In [2]:
```
# Leer la metadata asociada al dataset
with open("house-votes-84.names") as f:
    #print(f.read())
    (f.read())
```

In [3]:
```
#Importamos los archivos contenidos en el folder y chequeamos los data
dfHouseVotes = pd.read_csv('house-votes-84.data', sep=',')
```

De acuerdo a la metadata tomada del repositorio, vamos a proceder con renombrar los headers del dataset:

In [4]:
```
# Definimos el listado de columnas y renombramos los headers del datas
dfHouseVotes.columns = ['Class Name','handicapped-infants','water-proj
                        'adoption-of-the-budget-resolution','physician
                        'religious-groups-in-schools','anti-satellite-
                        'mx-missile','immigration','synfuels-corporati
                        'superfund-right-to-sue','crime','duty-free-ex
                        'export-administration-act-south-africa']
```

In [5]:
```
# chequeamos de nuevo
dfHouseVotes.tail(5)
```

Out[5]:

	Class Name	handicapped-infants	water-project-cost-sharing	adoption-of-the-budget-resolution	physician-fee-freeze	el-salvador-aid	religious-groups-in-schools	anti-satellite-test-ban
429	republican	n	n	y	y	y	y	n
430	democrat	n	n	y	n	n	n	y
431	republican	n	?	n	y	y	y	n
432	republican	n	n	n	y	y	y	?
433	republican	n	y	n	y	y	y	n

Ahora, vamos a realizar validaciones generales como número de columnas, cantidad de valores nulos, el tipo de variable de las columnas, distribuciones, gráficas entre otros:

In [6]:
```
print('Total filas y columnas: {}'.format(dfHouseVotes.shape))
```

```
Total filas y columnas: (434, 17)
```

In [7]:
```
# Ciclo para ver el número de registros por feature
for i in (dfHouseVotes.columns):
    print('Columna (feature) {} y número de registros: {}'.format(i, d
```

...

In [8]:
```
# Número de valores NA
for i in (dfHouseVotes.columns):
    print('Columna (feature) {} y número de registros NA: {}'.format(i
```

```
Columna (feature) Class Name y número de registros NA: 0
Columna (feature) handicapped-infants y número de registros NA: 0
Columna (feature) water-project-cost-sharing y número de registros NA
: 0
Columna (feature) adoption-of-the-budget-resolution y número de regis
tros NA: 0
Columna (feature) physician-fee-freeze y número de registros NA: 0
Columna (feature) el-salvador-aid y número de registros NA: 0
Columna (feature) religious-groups-in-schools y número de registros N
A: 0
Columna (feature) anti-satellite-test-ban y número de registros NA: 0
Columna (feature) aid-to-nicaraguan-contras y número de registros NA:
0
Columna (feature) mx-missile y número de registros NA: 0
Columna (feature) immigration y número de registros NA: 0
Columna (feature) synfuels-corporation-cutback y número de registros
NA: 0
Columna (feature) education-spending y número de registros NA: 0
Columna (feature) superfund-right-to-sue y número de registros NA: 0
Columna (feature) crime y número de registros NA: 0
Columna (feature) duty-free-exports y número de registros NA: 0
Columna (feature) export-administration-act-south-africa y número de
registros NA: 0
```

In [9]:
```
# Conteo de valores nulos por feature:
dfHouseVotes.isnull().sum()
```

Out[9]:
```
Class Name                              0
handicapped-infants                     0
water-project-cost-sharing              0
adoption-of-the-budget-resolution       0
physician-fee-freeze                    0
el-salvador-aid                         0
religious-groups-in-schools             0
anti-satellite-test-ban                 0
aid-to-nicaraguan-contras               0
mx-missile                              0
immigration                             0
synfuels-corporation-cutback            0
education-spending                      0
superfund-right-to-sue                  0
crime                                   0
duty-free-exports                       0
export-administration-act-south-africa  0
dtype: int64
```

In [10]:
```
# Estructura general de las columnas y sus tipos de datos:
dfHouseVotes.info()
```

...

En términos generales, vemos que la data no tiene inconsistencias frente a valor nulos y sus descripciones son claras para el entendimiento del ejercicio.

Ahora, revisaremos algunas gráficas y concentraciones tanto para el partido republicano como para el democráta:

In [11]:
```
# Revisamos la distribución y porcentaje de senadores por partido polí
dfHouseVotes.groupby('Class Name').count()
```

Out[11]:

Class Name	handicapped-infants	water-project-cost-sharing	adoption-of-the-budget-resolution	physician-fee-freeze	el-salvador-aid	religious-groups-in-schools	anti-satellite-test-ban	nicar
democrat	267	267	267	267	267	267	267	
republican	167	167	167	167	167	167	167	

In [12]:
```
# Calculamos porcentaje de distribución:
dfHouseVotes.groupby('Class Name')['handicapped-infants'].count() / df
```

Out[12]:
```
Class Name
democrat      0.615207
republican    0.384793
Name: handicapped-infants, dtype: float64
```

In [13]:
```
# Conteo general por tipo de decisión:
for element in range(len(dfHouseVotes.columns)):
    #print(round(dfHouseVotes.groupby(['Class Name', dfHouseVotes.colu
    (round(dfHouseVotes.groupby(['Class Name', dfHouseVotes.columns[el
```

In [14]:
```
# porcentaje general por tipo de decisión:
for element in range(len(dfHouseVotes.columns)):
    #print(round(dfHouseVotes.groupby([dfHouseVotes.columns[element]])
    #     / dfHouseVotes[dfHouseVotes.columns[element]].count(), 2)
    #     )
    (round(dfHouseVotes.groupby([dfHouseVotes.columns[element]])[dfHou
         / dfHouseVotes[dfHouseVotes.columns[element]].count(), 2)
         )
```

Teniendo en cuenta las últimas 4 líneas de código podemos confirmar las siguientes observaciones:

A nivel de partidos:

- En 1984 la mayoría política fue el partido demócrata con 267 congresistas (61.5%) frente a 167 del partido republicano (38.5%) lo cual se podrá reflejar en un resumen general de aprobación;

- El partido demócrata tuvo absoluta, relativa o parcial mayoría de aprobación para los siguientes temas tratados a nivel de partido:
  - handicapped-infants (35%)
  - water-project-cost-sharing (27%)
  - adoption-of-the-budget-resolution (53%)
  - anti-satellite-test-ban (46%)
  - aid-to-nicaraguan-contras (50%)
  - mx-missile (43%)
  - immigration (28%)
  - duty-free-exports (37%)
  - export-administration-act-south-africa (39%)

- El partido Republicano tuvo absoluta o parcial mayoría para los siguientes temas tratados:
  - el-salvador-aid (36%)
  - religious-groups-in-schools (34%)
  - education-spending (30%)
  - superfund-right-to-sue (31%)

A nivel general:

- A nivel general, es decir, sin tener en cuenta partido político, el 43% de las decisiones fueron positivas, el 54% negativas y tan sólo el 3% fue abstención.

- Las propuestas con porcentajes casi de empate entre el sí y el no fueron las siguientes:
  - water-project-cost-sharing
  - el-salvador-aid
  - mx-missile
  - immigration
  - superfund-right-to-sue

Esto significa que a pesar de que los demócratas cuenten con mayorías en el congreso estos proyectos generan profunda división a nivel interno y general.

- A nivel general, las propuestas que ganaron amplia aceptación fueron:
  - adoption-of-the-budget-resolution (58%)
  - religious-groups-in-schools (62%)
  - anti-satellite-test-ban (55%)
  - aid-to-nicaraguan-contras (56%)
  - crime (57%)
  - export-administration-act-south-africa (62%)

  La mayoría de estos temas están relacionado con temas de interés nacional, por lo cual tienen más posibilidad de aceptación. Ahora revisemos los proyectos que fueron rechazados:

  - handicapped-infants (54%)
  - physician-fee-freeze (57%)
  - synfuels-corporation-cutback (61%)
  - education-spending (54%)
  - duty-free-exports (53%)

  Se resalta el hecho de que proyectos de inversión social como educación y ayuda a personas minusválidas haya generado total rechazo hacia estos respectivos proyectos .En el caso de abstención se generaron votos por debajo del 11% en general, solamente el proyecto export-administration-act-south-africa tuvo una abstención significativa con un 24% aproximadamente.

  A continuación vamos a realizar algunos gráficos para ver concentraciones y tendencias, entre otros.

In [15]:
```
# Vamos a revisar la distribución en general por columnas para ver las
sns.set()
dfHouseVotes_list = ['Class Name','handicapped-infants','water-project
'el-salvador-aid','religious-groups-in-schools','anti-satellite-test-b
'synfuels-corporation-cutback','education-spending','superfund-right-t

for j in dfHouseVotes_list:
    print(j)
    plt.figure(figsize=(4, 3))
    plt.title(j, fontweight ="bold")
    dfHouseVotes[j].hist(bins=10, color = 'skyblue', histtype ='bar')
    plt.grid(visible = None)
    plt.show()
```

...

2. Análisis exploratorio de datos dataset adults.data

El objectivo de esta segunda parte es realizar el análisis exploratorio de datos para el dataset **adults** el cual se puede encontrar en la siguiente página web (y dentro de este folder también)y el cual se encuentra referenciado al final de este notebook:

- https://archive.ics.uci.edu/ml/datasets/adult (https://archive.ics.uci.edu/ml/datasets/adult)

Además, es importante tener en cuenta que dentro de este análisis se aplicarán varias funciones de Python utilizadas en el primer ejercicio para mayor practicidad.

Las siguientes descripciones serán útiles para conocer la estructura de los datasets:

Descripción general:

Según el repositorio oficial, el dataset contiene una muestra de datos del censo nacional de Estados Unidos de 1994 para crear prediciones relacionadas con si una persona podría ganar más de 50K dólares al año. Los features contenidos son los siguientes: :

1. age
2. workclass
3. fnlwgt
4. education
5. education-number
6. marital-status
7. occupation
8. relationship
9. race
10. sex
11. capital-gain
12. capital-loss
13. hours-per-week
14. native-country
15. Gana más de > 50K, menos de <=K (target variable)

A simple vista, podemos ver una mezcla entre variables categóricas y numéricas. Con todo lo explicado anteriormente, vamos a revisar la estructura general de los datos:

In [16]:
```
# Leemos el dataset y reajustamos columnas:
dfAdults = pd.read_csv('adult.data', sep=',')
```

In [17]:
```
# Metadata del dataset
with open("adult.names") as f:
    (f.read())
    #print(f.read())
```

De acuerdo a la metadata tomada del repositorio, vamos a proceder con renombrar los headers del dataset:

In [18]:
```
# Renombramos las columnas:
dfAdults.columns = ['age','workclass','fnlwgt', 'education','education
'relationship','race','sex','capital-gain','capital-loss','hours-per-w
]
```

In [19]:
```
# Chequeamos la estructura general ajustada del dataset:
dfAdults.tail(1)
```

Out[19]:

	age	workclass	fnlwgt	education	education-number	marital-status	occupation	relationship	rac
32559	52	Self-emp-inc	287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	Whit

Ahora, vamos a realizar validaciones generales como número de columnas, cantidad de valores nulos, el tipo de variable de las columnas, distribuciones, gráficas entre otros:

In [20]:
```
# Dimensión del dataset
print('Total filas y columnas: {}'.format(dfAdults.shape))
```

```
Total filas y columnas: (32560, 15)
```

In [21]:
```
# Número de registros por feature
for i in (dfAdults.columns):
    print('Columna (feature) {} y número de registros: {}'.format(i, d
```

```
Columna (feature) age y número de registros: 32560
Columna (feature) workclass y número de registros: 32560
Columna (feature) fnlwgt y número de registros: 32560
Columna (feature) education y número de registros: 32560
Columna (feature) education-number y número de registros: 32560
Columna (feature) marital-status y número de registros: 32560
Columna (feature) occupation y número de registros: 32560
Columna (feature) relationship y número de registros: 32560
Columna (feature) race y número de registros: 32560
Columna (feature) sex y número de registros: 32560
Columna (feature) capital-gain y número de registros: 32560
Columna (feature) capital-loss y número de registros: 32560
Columna (feature) hours-per-week y número de registros: 32560
Columna (feature) native-country y número de registros: 32560
Columna (feature) salary-condition y número de registros: 32560
```

In [22]:
```
# conteo de registros NA
for i in (dfAdults.columns):
    print('Columna (feature) {} y número de registros NA: {}'.format(i
```

```
Columna (feature) age y número de registros NA: 0
Columna (feature) workclass y número de registros NA: 0
Columna (feature) fnlwgt y número de registros NA: 0
Columna (feature) education y número de registros NA: 0
Columna (feature) education-number y número de registros NA: 0
Columna (feature) marital-status y número de registros NA: 0
Columna (feature) occupation y número de registros NA: 0
Columna (feature) relationship y número de registros NA: 0
Columna (feature) race y número de registros NA: 0
Columna (feature) sex y número de registros NA: 0
Columna (feature) capital-gain y número de registros NA: 0
Columna (feature) capital-loss y número de registros NA: 0
Columna (feature) hours-per-week y número de registros NA: 0
Columna (feature) native-country y número de registros NA: 0
Columna (feature) salary-condition y número de registros NA: 0
```

In [23]:
```
# Tipo de variables del dataset:
dfAdults.info()
```

...

In [24]:
```
# Para las columnas categóricas, revisamos los valores únicos:
list_categorical_features = ['workclass', 'education', 'marital-status
                             'race', 'sex', 'native-country', 'salary-
for m in list_categorical_features:
    print(dfAdults[m].unique())
```

```
[' Self-emp-not-inc' ' Private' ' State-gov' ' Federal-gov' ' Local-g
ov'
 ' ?' ' Self-emp-inc' ' Without-pay' ' Never-worked']
[' Bachelors' ' HS-grad' ' 11th' ' Masters' ' 9th' ' Some-college'
 ' Assoc-acdm' ' Assoc-voc' ' 7th-8th' ' Doctorate' ' Prof-school'
 ' 5th-6th' ' 10th' ' 1st-4th' ' Preschool' ' 12th']
[' Married-civ-spouse' ' Divorced' ' Married-spouse-absent'
 ' Never-married' ' Separated' ' Married-AF-spouse' ' Widowed']
[' Exec-managerial' ' Handlers-cleaners' ' Prof-specialty'
 ' Other-service' ' Adm-clerical' ' Sales' ' Craft-repair'
 ' Transport-moving' ' Farming-fishing' ' Machine-op-inspct'
 ' Tech-support' ' ?' ' Protective-serv' ' Armed-Forces'
 ' Priv-house-serv']
[' Husband' ' Not-in-family' ' Wife' ' Own-child' ' Unmarried'
 ' Other-relative']
[' White' ' Black' ' Asian-Pac-Islander' ' Amer-Indian-Eskimo' ' Othe
r']
[' Male' ' Female']
[' United-States' ' Cuba' ' Jamaica' ' India' ' ?' ' Mexico' ' South'
 ' Puerto-Rico' ' Honduras' ' England' ' Canada' ' Germany' ' Iran'
 ' Philippines' ' Italy' ' Poland' ' Columbia' ' Cambodia' ' Thailand
'
 ' Ecuador' ' Laos' ' Taiwan' ' Haiti' ' Portugal' ' Dominican-Republ
ic'
 ' El-Salvador' ' France' ' Guatemala' ' China' ' Japan' ' Yugoslavia
'
 ' Peru' ' Outlying-US(Guam-USVI-etc)' ' Scotland' ' Trinadad&Tobago'
 ' Greece' ' Nicaragua' ' Vietnam' ' Hong' ' Ireland' ' Hungary'
 ' Holand-Netherlands']
[' <=50K' ' >50K']
```

De acuerdo a la celda anterior, las categorías workclass y native-country tiene para aquellas personas que no indicaron de forma el valor **?**

In [25]:
```
dfAdults['workclass'].value_counts()
```

Out[25]:
```
 Private             22696
 Self-emp-not-inc     2541
 Local-gov            2093
 ?                    1836
 State-gov            1297
 Self-emp-inc         1116
 Federal-gov           960
 Without-pay            14
 Never-worked            7
Name: workclass, dtype: int64
```

In [26]:
```
dfAdults['native-country'].value_counts()
```

Out[26]:
```
 United-States                 29169
 Mexico                          643
 ?                               583
 Philippines                     198
 Germany                         137
 Canada                          121
 Puerto-Rico                     114
 El-Salvador                     106
 India                           100
 Cuba                             95
 England                          90
 Jamaica                          81
 South                            80
 China                            75
 Italy                            73
 Dominican-Republic               70
 Vietnam                          67
 Guatemala                        64
 Japan                            62
 Poland                           60
```

In [27]:
```
dfAdults.isnull().sum()
```

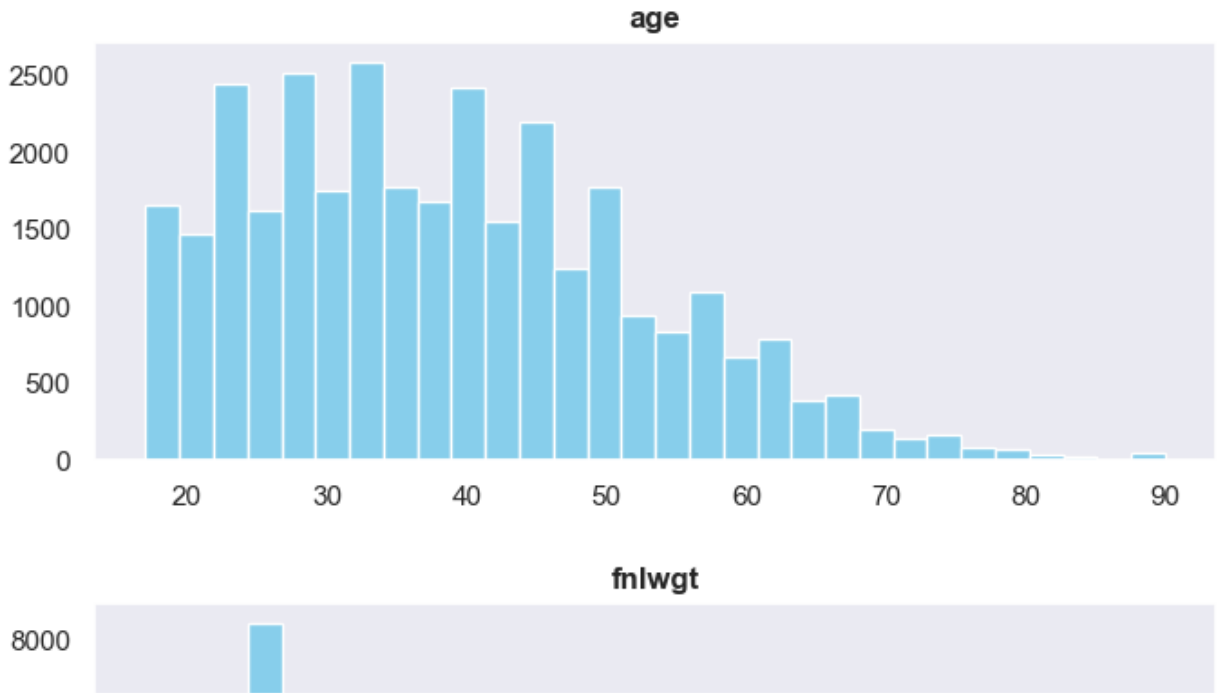
Out[27]:
```
age                0
workclass          0
fnlwgt             0
education          0
education-number   0
marital-status     0
occupation         0
relationship       0
race               0
sex                0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     0
salary-condition   0
dtype: int64
```

In [28]:
```
dfAdults.isna().sum()
```

Out[28]:
```
age                0
workclass          0
fnlwgt             0
education          0
education-number   0
marital-status     0
occupation         0
relationship       0
race               0
sex                0
capital-gain       0
capital-loss       0
hours-per-week     0
native-country     0
salary-condition   0
dtype: int64
```

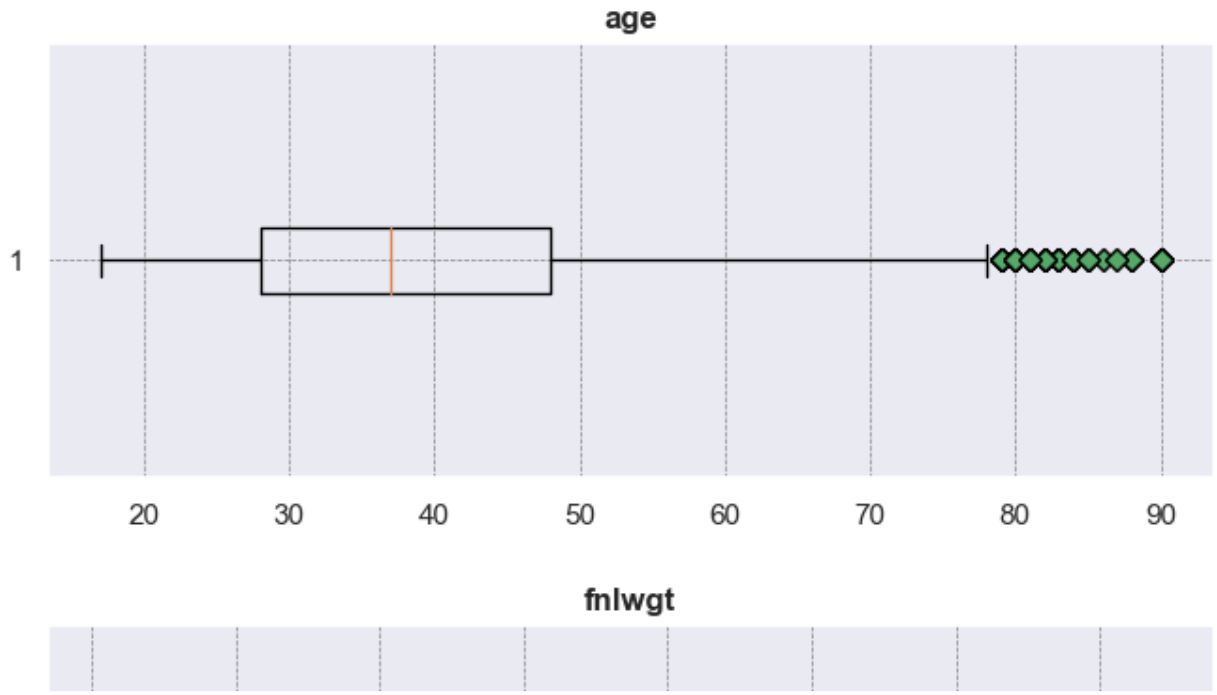
En general vemos que los valores tipo **?** tienen una presencia significativa en general, sin embargo, se considera mantener los datos en caso de una eventual aplicación de algún modelo supervisado. Además no se evidencian valores NA o nulos, por lo cual no se requieren tareas de data cleaning.
Ahora, vamos a gráficamente la distribución de columnas numêricas y dispersiones a nivel general:

In [29]:
```
list_numerical_features = ['age', 'fnlwgt', 'education-number', 'capit
for j in list_numerical_features:
    plt.figure(figsize=(8, 3))
    plt.title(j, fontweight ="bold")
    dfAdults[j].hist(bins=30, color = 'skyblue', histtype ='bar')
    plt.grid(visible = None)
    plt.show()
```



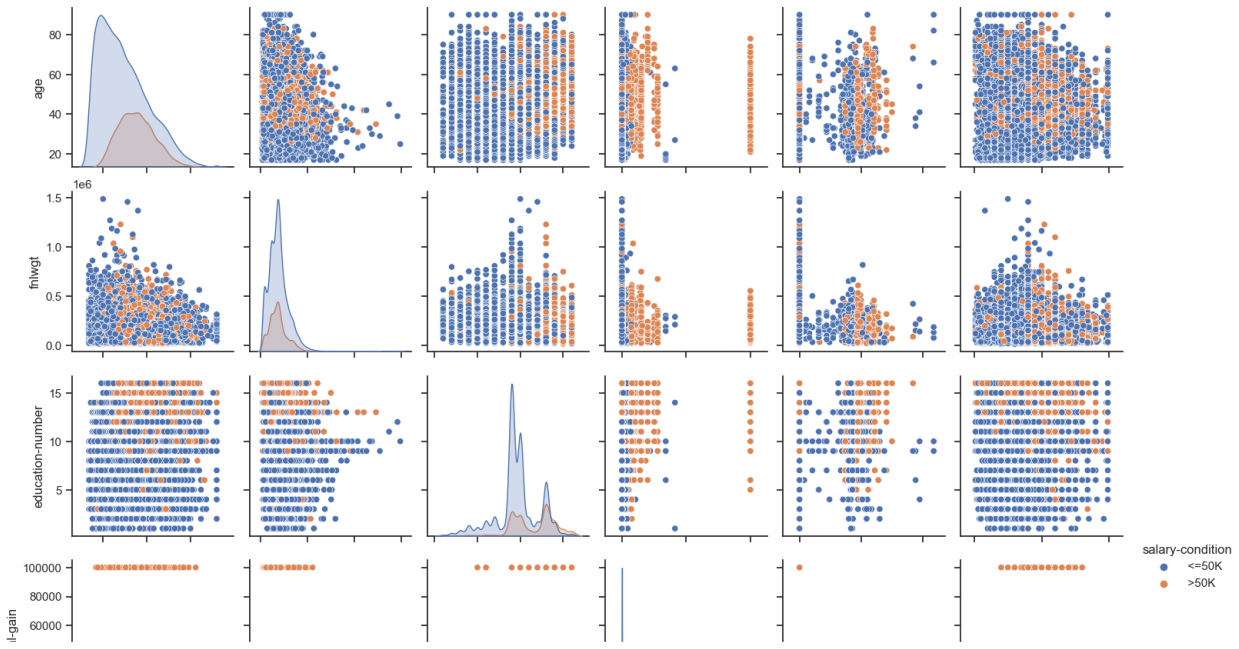
In [30]:
```
green_diamond = dict(markerfacecolor = 'g', marker='D')

for j in list_numerical_features:
    plt.figure(figsize=(8, 3))
    plt.title(j, fontweight ="bold")
    plt.boxplot(dfAdults[j], flierprops = green_diamond, vert= False)
    plt.grid(color = 'gray', linestyle = '--', linewidth = 0.5)
    plt.show()
```



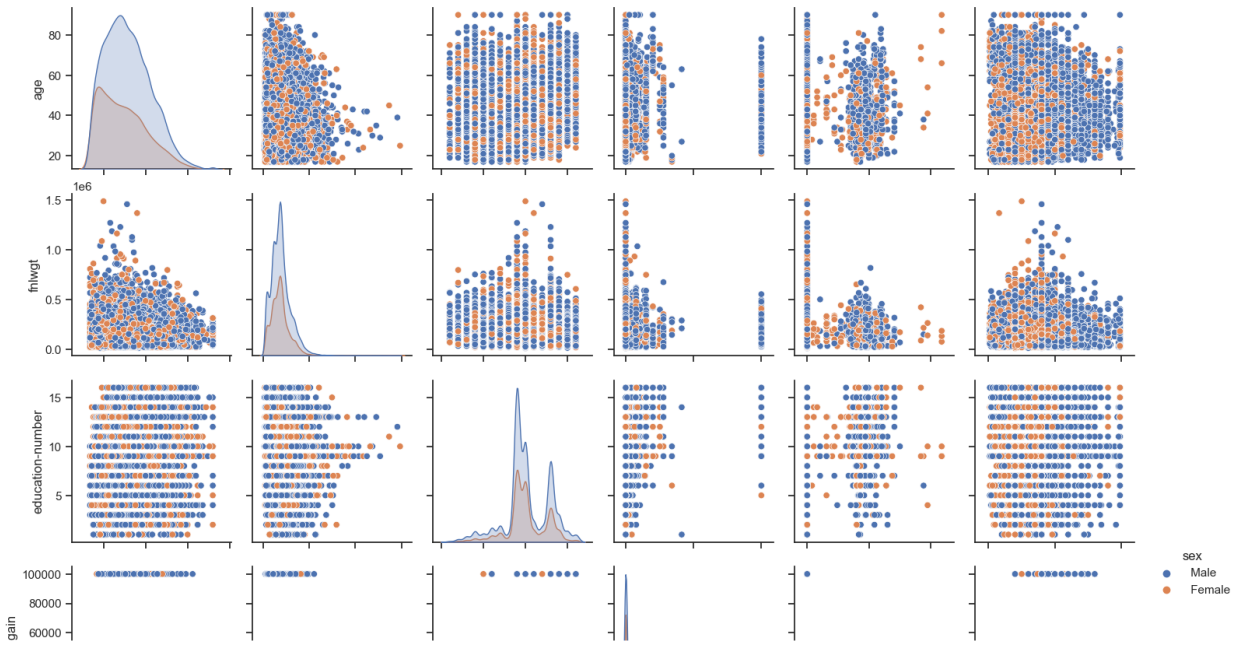
In [31]:
```
sns.set_theme(style="ticks")
sns.pairplot(dfAdults, hue="salary-condition")
```

Out[31]: <seaborn.axisgrid.PairGrid at 0x7fcbc03aaa00>



In [32]:
```
sns.set_theme(style="ticks")
sns.pairplot(dfAdults, hue="sex")
```

Out[32]:
```
<seaborn.axisgrid.PairGrid at 0x7fcbb2257580>
```



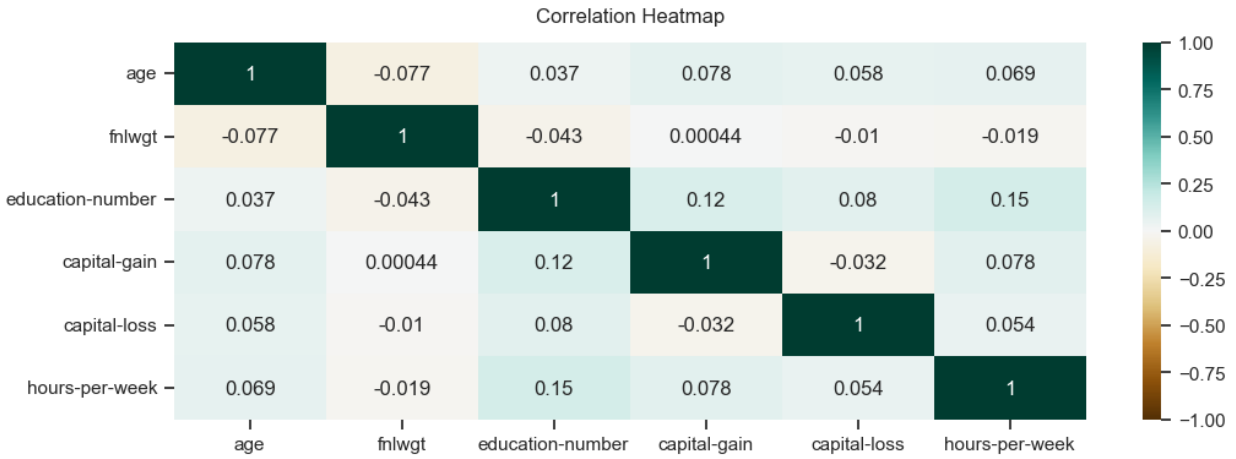
Se evidencia en términos generales que la mayoría de la muestra poblacional que participó en el censo gana por debajo de los 50K anuales.

De acuerdo a los resultados anteriores, se tienen las siguientes observaciones:

* Las variables age y fnwlt tienen distribuciones no sesgadas, mien tras que resto presenta ciertas concentraciones.
* en lox boxplots vemos una gran cantidad de valores atípicos, por lo cual se deben tener en cuenta al considerar que modelo se puede aplicar para evitar baja calidad.
* En el primer panel de gráficos, se evidencia que gran porcentaje de la población gana por debajo de 50K USD.
* La población tiene más concentración en hombres que mujeres por e dad y en horas de trabajo una parte significativa de hombres trabaj a más de 45 horas semanales.

Ahora veremos la correlación entre todas las variables numéricas:

In [35]:
```
plt.figure(figsize=(12, 4))
heatmap = sns.heatmap(dfAdults.corr(), vmin=-1, vmax=1, annot=True, cm
heatmap.set_title('Correlation Heatmap', fontdict={'fontsize':12}, pad
```



De acuerdo a los resultados anteriores, existen relaciones no tan fuertes a nivel general que se pueden profundizar entre las siguientes variables:

* hours-per-week y education-number
* capital-gain y education-number

Conclusiones

- A nivel general realizamos un análisis exploratorio para generar visualizaciones y conclusiones relevantes con dos datasets para entender el uso de herramientas y técnicas previas en construcción de modelos.

Referencias:

[1]. Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml] (http://archive.ics.uci.edu/ml%5D). Irvine, CA: University of California, School of Information and Computer Science.

[2]. Congressional Voting Records. Kaggle. Taken from: https://www.kaggle.com/datasets/devvret/congressional-voting-records (https://www.kaggle.com/datasets/devvret/congressional-voting-records)

[3]. Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[4]. J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.