In [9]:
```
import pandas as pd
import numpy as np

df_car_prices = pd.read_csv(
    filepath_or_buffer = "D:\\output\\cars_prepared.csv",
    engine = 'pyarrow'
)
```

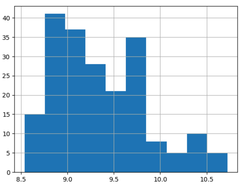
In [10]:
```
list_reduced = [
'symboling',
'doornumber',
'wheelbase',
'carlength',
'carwidth',
'carheight',
'curbweight',
'cylindernumber',
'enginesize',
'boreratio',
'stroke',
'compressionratio',
'horsepower',
'peakrpm',
'citympg',
'highwaympg'
]

X = df_car_prices[list_reduced]
y = df_car_prices['price']
X_Train = df_car_prices.loc[df_car_prices['TrainTest'] == 'Train'][list_reduced]
y_Train = df_car_prices.loc[df_car_prices['TrainTest'] == 'Train','price']
```

In [11]:
```
# Gaussian regression, vanilla linear regression
```

In [12]:
```
df_car_prices['price'].hist()
from sklearn.linear_model import LinearRegression
LinearRegression_price = LinearRegression().fit(
    X = X_Train,
    y = y_Train
)
[LinearRegression_price.score(
    X = X.loc[df_car_prices['TrainTest'] == j],
    y = y.loc[df_car_prices['TrainTest'] == j]
) for j in ['Train','Validation','Test']]
```

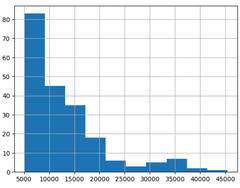
Out[12]:
```
[0.9168924662244572, 0.8488000838419207, 0.7694823388568295]
```



Gamma regression

In [13]:
```
(np.e**df_car_prices['price']).hist()
from sklearn.linear_model import GammaRegressor
GammaRegressor_price = GammaRegressor().fit(
    X = X_Train,
    y = np.e**y_Train
)
GammaRegressor_price.score(
    X = X_Train,
    y = np.e**y_Train
)
[GammaRegressor_price.score(
    X = X.loc[df_car_prices['TrainTest'] == j],
    y = (np.e**y).loc[df_car_prices['TrainTest'] == j]
) for j in ['Train','Validation','Test']]
```

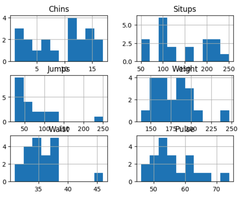
Out[13]:
```
[np.float64(0.8546796695923717),
 np.float64(0.8404356340723973),
 np.float64(0.8248977622262407)]
```



Poisson regression

In [14]:
```
from sklearn.datasets import load_linnerud
df_linnerud = pd.concat(
    [obj4 = load_linnerud(
        return_X_y=True,
        as_frame=True
    ),
    axis = 1
)
df_linnerud.hist()
from sklearn.linear_model import PoissonRegressor
PoissonRegressor_Chins = PoissonRegressor(1.0).fit(
    X = df_linnerud[['Weight', 'Waist', 'Pulse']],
    y = df_linnerud['Situps']
)
PoissonRegressor_Chins.score(
    X = df_linnerud[['Weight', 'Waist', 'Pulse']],
    y = df_linnerud['Situps']
)
```

Out[14]:
```
np.float64(0.49891538830585636)
```



Tweedie regression

In [15]:
```
from sklearn.linear_model import TweedieRegressor
array_power = np.linspace(0,4,21)
pd.DataFrame([{
    'power' : array_power,
    'score' : [TweedieRegressor(power=power).fit(
        X = X_Train,
        y = y_Train
    ).score(
        X = X.loc[df_car_prices['TrainTest'] == 'Validation'],
        y = y.loc[df_car_prices['TrainTest'] == 'Validation']
    ) for power in array_power]
}])
```

Out[15]:

	power	score
0	0.0	0.830648
1	0.2	0.852851
2	0.4	0.852382
3	0.6	0.851766
4	0.8	0.851157
5	1.0	0.850654
6	1.2	0.850431
7	1.4	0.849907
8	1.6	0.848048
9	1.8	0.843151
10	2.0	0.832629
11	2.2	0.812368
12	2.4	0.776322
13	2.6	0.718166
14	2.8	0.632749
15	3.0	0.518132
16	3.2	0.395656
17	3.4	0.280805
18	3.6	0.186160
19	3.8	0.112224
20	4.0	0.058145

In [16]:
```
TweedieRegressor_SalePrice = TweedieRegressor(power = 0.8,max_iter = 1000).fit(
    X = X_Train,
    y = y_Train
)
[TweedieRegressor_SalePrice.score(
    X = X.loc[df_car_prices['TrainTest'] == j],
    y = y.loc[df_car_prices['TrainTest'] == j]
) for j in ['Train','Validation','Test']]
```

Out[16]:
```
[np.float64(0.9103668865381285),
 np.float64(0.8511571798019193),
 np.float64(0.8123857986961677)]
```

Logistic regression

In [17]:
```
df_loan = pd.read_csv(
    filepath_or_buffer = "D:\\loan_data.csv",
    engine = 'pyarrow'
)
df_loan
```

Out[17]:

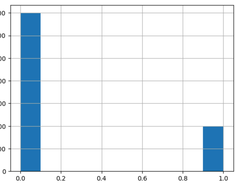
	loan_status	partition	person_education_2_moderate	person_education_3_high	person_home_ownership_3_high	loan_intent_2_moderate	loan_intent_3_high
0	1	test	0	0	1	1	0
1	0	test	1	0	0	0	0
2	1	train	1	0	0	0	1
3	1	train	0	1	1	0	1
4	1	test	0	0	1	0	1
...	...	...	...	...	...	...	...
44995	1	test	0	0	1	0	1
44996	1	validation	0	0	1	1	0
44997	1	train	0	0	1	0	1
44998	1	train	0	1	1	0	0
44999	1	train	1	0	1	0	1

45000 rows × 7 columns

In [19]:
```
list_predictors = [
    'person_education_2_moderate',
    'person_education_3_high',
    'person_home_ownership_3_high',
    'loan_intent_2_moderate',
    'loan_intent_3_high',
]
X = df_loan[list_predictors]
y = df_loan['loan_status']
X_Train = df_loan.loc[df_loan['partition'] == 'train'][list_predictors]
y_Train = df_loan.loc[df_loan['partition'] == 'train','loan_status']
```

In [20]:
```
df_loan['loan_status'].hist()
from sklearn.linear_model import LogisticRegression
LogisticRegression_class = LogisticRegression().fit(
    X = X_Train,
    y = y_Train
)
[LogisticRegression_class.score(
    X = X.loc[df_loan['partition'] == j],
    y = y.loc[df_loan['partition'] == j]
) for j in ['train','validation','test']]
```

Out[20]:
```
[0.7716262975778547, 0.7839994635193133, 0.7777556440903054]
```


35000
30000
25000
20000
15000
10000
5000
0
0.0
0.2
0.4
0.6
0.8
1.0