In [2]:**import** **numpy** **as** **np**

**import** **pandas** **as** **pd**

**import** **matplotlib.pyplot** **as** **plt**

**import** **pandas\_profiling**

**import** **seaborn** **as** **sns**

%matplotlib inline

In [3]:

hep\_df=pd.read\_csv("hepatitis\_csv.csv")

In [4]:

hep\_df

In [30]:

hep\_df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 155 entries, 0 to 154

Data columns (total 20 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 age 155 non-null int64

1 sex 155 non-null object

2 steroid 155 non-null bool

3 antivirals 155 non-null bool

4 fatigue 155 non-null bool

5 malaise 155 non-null bool

6 anorexia 155 non-null bool

7 liver\_big 145 non-null object

8 liver\_firm 144 non-null object

9 spleen\_palpable 150 non-null object

10 spiders 150 non-null object

11 ascites 150 non-null object

12 varices 150 non-null object

13 bilirubin 149 non-null float64

14 alk\_phosphate 126 non-null float64

15 sgot 151 non-null float64

16 albumin 139 non-null float64

17 protime 88 non-null float64

18 histology 155 non-null bool

19 class 155 non-null object

dtypes: bool(6), float64(5), int64(1), object(8)

memory usage: 18.0+ KB

In [12]:

hep\_df.isnull().sum()

Out[12]:

age 0

sex 0

steroid 1

antivirals 0

fatigue 1

malaise 1

anorexia 1

liver\_big 10

liver\_firm 11

spleen\_palpable 5

spiders 5

ascites 5

varices 5

bilirubin 6

alk\_phosphate 29

sgot 4

albumin 16

protime 67

histology 0

class 0

dtype: int64

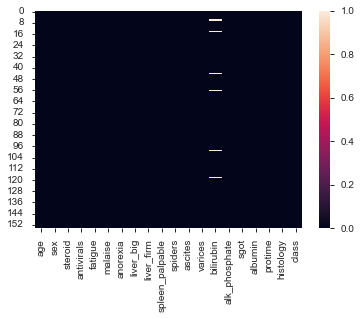
In [66]:

*# \*\*to check missing values\*\**

sns.heatmap(hep\_df.isnull())

Out[66]:

<AxesSubplot:>



In [21]:

hep\_df["steroid"].fillna(hep\_df["steroid"].mode()[0], inplace = **True**)

In [22]:

hep\_df["fatigue"].fillna(hep\_df["fatigue"].mode()[0], inplace = **True**)

In [23]:

hep\_df["malaise"].fillna(hep\_df["malaise"].mode()[0], inplace = **True**)

In [24]:

hep\_df["anorexia"].fillna(hep\_df["anorexia"].mode()[0], inplace = **True**)

In [34]:

hep\_df.isnull().sum()

Out[34]:

age 0

sex 0

steroid 0

antivirals 0

fatigue 0

malaise 0

anorexia 0

liver\_big 0

liver\_firm 11

spleen\_palpable 5

spiders 5

ascites 5

varices 5

bilirubin 6

alk\_phosphate 29

sgot 4

albumin 16

protime 67

histology 0

class 0

dtype: int64

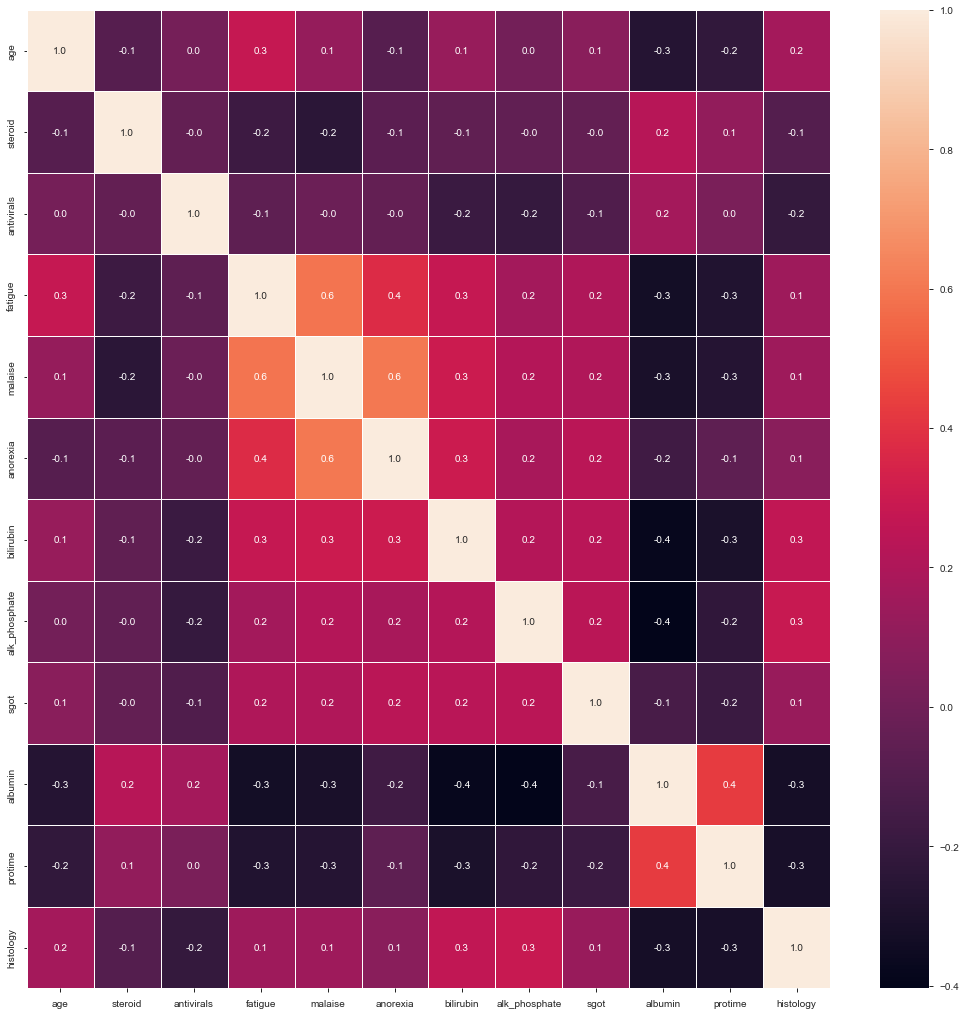
In [26]:

f,ax=plt.subplots(figsize = (18,18))

sns.heatmap(hep\_df.corr(method="pearson"),annot= **True**,linewidths=0.5,fmt = ".1f",ax=ax)

Out[26]:

<AxesSubplot:>



In [33]:

hep\_df["liver\_big"].fillna(hep\_df["liver\_big"].mode()[0], inplace = **True**)

In [35]:

hep\_df["liver\_firm"].fillna(hep\_df["liver\_firm"].mode()[0], inplace = **True**)

In [ ]:

In [38]:

hep\_df["spleen\_palpable"].fillna(hep\_df["spleen\_palpable"].mode()[0], inplace = **True**)

In [40]:

hep\_df["spiders"].fillna(hep\_df["spiders"].mode()[0], inplace = **True**)

In [41]:

hep\_df["ascites"].fillna(hep\_df["ascites"].mode()[0], inplace = **True**)

In [42]:

hep\_df["varices"].fillna(hep\_df["varices"].mode()[0], inplace = **True**)

In [70]:

hep\_df.isnull().sum()

Out[70]:

age 0

sex 0

steroid 0

antivirals 0

fatigue 0

malaise 0

anorexia 0

liver\_big 0

liver\_firm 0

spleen\_palpable 0

spiders 0

ascites 0

varices 0

bilirubin 6

alk\_phosphate 0

sgot 0

albumin 0

protime 0

histology 0

class 0

dtype: int64

In [45]:

hep\_df["alk\_phosphate"].median()

Out[45]:

85.0

In [46]:

hep\_df["alk\_phosphate"].mean()

Out[46]:

105.32539682539682

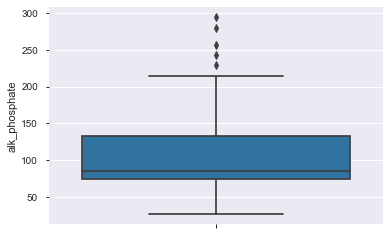
In [47]:

*#box plot to check outlier*

sns.boxplot( y=hep\_df["alk\_phosphate"])

Out[47]:

<AxesSubplot:ylabel='alk\_phosphate'>



In [48]:

hep\_df["alk\_phosphate"].fillna(hep\_df["alk\_phosphate"].median(), inplace = **True**)

In [50]:

hep\_df["sgot"]

Out[50]:

0 18.0

1 42.0

2 32.0

3 52.0

4 200.0

...

150 242.0

151 142.0

152 20.0

153 19.0

154 19.0

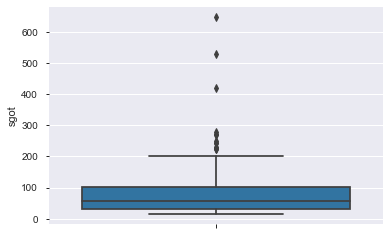
Name: sgot, Length: 155, dtype: float64

In [51]:

sns.boxplot( y=hep\_df["sgot"])

Out[51]:

<AxesSubplot:ylabel='sgot'>



In [52]:

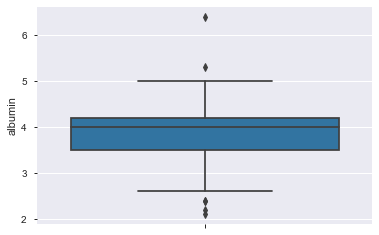
hep\_df["sgot"].fillna(hep\_df["sgot"].median(), inplace = **True**)

In [59]:

sns.boxplot( y=hep\_df["albumin"])

Out[59]:

<AxesSubplot:ylabel='albumin'>



In [60]:

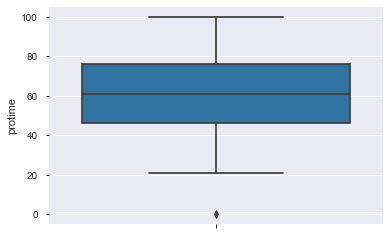
hep\_df["albumin"].fillna(hep\_df["albumin"].median(), inplace = **True**)

In [62]:

sns.boxplot( y=hep\_df["protime"])

Out[62]:

<AxesSubplot:ylabel='protime'>

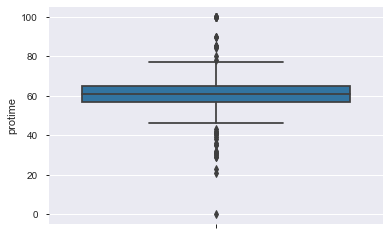


In [64]:

sns.boxplot( y=hep\_df["protime"])

Out[64]:

<AxesSubplot:ylabel='protime'>



In [63]:

*#as there are more outling treating outiers*

hep\_df["protime"].fillna(hep\_df["protime"].median(), inplace = **True**)

In [71]:

Q1 = hep\_df["protime"].quantile(0.25)

Q3 = hep\_df["protime"].quantile(0.75)

IQR = Q3 - Q1

In [76]:

print(hep\_df["protime"] < (Q1 - 1.5 \* IQR) |hep\_df["protime"] > (Q3 + 1.5 \* IQR))

---------------------------------------------------------------------------

TypeError Traceback (most recent call last)

F:\anaconda\_inst\lib\site-packages\pandas\core\ops\array\_ops.py in na\_logical\_op(x, y, op)

301 # (xint or xbool) and (yint or bool)

--> 302 result = op(x, y)

303 except TypeError:

F:\anaconda\_inst\lib\site-packages\pandas\core\roperator.py in ror\_(left, right)

55 def ror\_(left, right):

---> 56 return operator.or\_(right, left)

57

TypeError: ufunc 'bitwise\_or' not supported for the input types, and the inputs could not be safely coerced to any supported types according to the casting rule ''safe''

During handling of the above exception, another exception occurred:

ValueError Traceback (most recent call last)

F:\anaconda\_inst\lib\site-packages\pandas\core\ops\array\_ops.py in na\_logical\_op(x, y, op)

315 try:

--> 316 result = libops.scalar\_binop(x, y, op)

317 except (

F:\anaconda\_inst\lib\site-packages\pandas\\_libs\ops.pyx in pandas.\_libs.ops.scalar\_binop()

ValueError: Buffer dtype mismatch, expected 'Python object' but got 'double'

The above exception was the direct cause of the following exception:

TypeError Traceback (most recent call last)

~\AppData\Local\Temp/ipykernel\_1840/400661805.py in <module>

----> 1 print(hep\_df["protime"] < (Q1 - 1.5 \* IQR) |hep\_df["protime"] > (Q3 + 1.5 \* IQR))

F:\anaconda\_inst\lib\site-packages\pandas\core\generic.py in \_\_array\_ufunc\_\_(self, ufunc, method, \*inputs, \*\*kwargs)

2030 self, ufunc: np.ufunc, method: str, \*inputs: Any, \*\*kwargs: Any

2031 ):

-> 2032 return arraylike.array\_ufunc(self, ufunc, method, \*inputs, \*\*kwargs)

2033

2034 # ideally we would define this to avoid the getattr checks, but

F:\anaconda\_inst\lib\site-packages\pandas\core\arraylike.py in array\_ufunc(self, ufunc, method, \*inputs, \*\*kwargs)

251

252 # for binary ops, use our custom dunder methods

--> 253 result = maybe\_dispatch\_ufunc\_to\_dunder\_op(self, ufunc, method, \*inputs, \*\*kwargs)

254 if result is not NotImplemented:

255 return result

F:\anaconda\_inst\lib\site-packages\pandas\\_libs\ops\_dispatch.pyx in pandas.\_libs.ops\_dispatch.maybe\_dispatch\_ufunc\_to\_dunder\_op()

F:\anaconda\_inst\lib\site-packages\pandas\core\ops\common.py in new\_method(self, other)

67 other = item\_from\_zerodim(other)

68

---> 69 return method(self, other)

70

71 return new\_method

F:\anaconda\_inst\lib\site-packages\pandas\core\arraylike.py in \_\_ror\_\_(self, other)

72 @unpack\_zerodim\_and\_defer("\_\_ror\_\_")

73 def \_\_ror\_\_(self, other):

---> 74 return self.\_logical\_method(other, roperator.ror\_)

75

76 @unpack\_zerodim\_and\_defer("\_\_xor\_\_")

F:\anaconda\_inst\lib\site-packages\pandas\core\series.py in \_logical\_method(self, other, op)

5511 rvalues = extract\_array(other, extract\_numpy=True, extract\_range=True)

5512

-> 5513 res\_values = ops.logical\_op(lvalues, rvalues, op)

5514 return self.\_construct\_result(res\_values, name=res\_name)

5515

F:\anaconda\_inst\lib\site-packages\pandas\core\ops\array\_ops.py in logical\_op(left, right, op)

390 filler = fill\_int if is\_self\_int\_dtype and is\_other\_int\_dtype else fill\_bool

391

--> 392 res\_values = na\_logical\_op(lvalues, rvalues, op)

393 # error: Cannot call function of unknown type

394 res\_values = filler(res\_values) # type: ignore[operator]

F:\anaconda\_inst\lib\site-packages\pandas\core\ops\array\_ops.py in na\_logical\_op(x, y, op)

326 f"Cannot perform '{op.\_\_name\_\_}' with a dtyped [{x.dtype}] array "

327 f"and scalar of type [{typ}]"

--> 328 ) from err

329

330 return result.reshape(x.shape)

TypeError: Cannot perform 'ror\_' with a dtyped [float64] array and scalar of type [bool]

In [ ]: