In [1]:

```
import pandas as pd
```

In [2]:

```
df = pd.read_csv('Oxygen Dataset Final.csv')
```

In [3]:

```
df.head()
```

Out[3]:

	age	gender	spo2	pr	c/nc	oxy_flow
0	27	0	74.0	72.0	1.0	6.0
1	53	1	NaN	110.0	NaN	28.0
2	56	0	99.0	98.0	1.0	NaN
3	26	1	NaN	110.0	1.0	4.0
4	52	0	69.0	84.0	1.0	0.0

In [4]:

```
df.tail()
```

Out[4]:

	age	gender	spo2	pr	c/nc	oxy_flow
199995	17	0	99.0	NaN	1.0	28.0
199996	48	1	99.0	NaN	1.0	5.0
199997	24	0	99.0	110.0	1.0	23.0
199998	100	1	99.0	95.0	1.0	25.0
199999	22	1	99.0	82.0	0.0	32.0

In [5]:

```
df.shape
```

Out[5]:

```
(200000, 6)
```

In [6]:

```
df.columns
```

Out[6]:

```
Index(['age', 'gender', 'spo2', 'pr', 'c/nc', 'oxy_flow'], dtype='object')
```

In [7]:

```
df.duplicated().sum()
```

Out[7]:

```
23486
```

In [8]:

```
df = df.drop_duplicates()
```

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
age             0
gender          0
spo2        23115
pr          24769
c/nc        25191
oxy_flow    29689
dtype: int64
```

In [10]:

```
df_new = df.dropna()
```

In [11]:

```
df_new.isnull().sum()
```

Out[11]:

```
age         0
gender      0
spo2        0
pr          0
c/nc        0
oxy_flow    0
dtype: int64
```

In [12]:

```
df_new.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 90848 entries, 0 to 199999
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       90848 non-null  int64  
 1   gender    90848 non-null  int64  
 2   spo2      90848 non-null  float64
 3   pr        90848 non-null  float64
 4   c/nc      90848 non-null  float64
 5   oxy_flow  90848 non-null  float64
dtypes: float64(4), int64(2)
memory usage: 4.9 MB
```

In [13]:

```
df_new.describe()
```

Out[13]:

	age	gender	spo2	pr	c/nc	oxy_flow
count	90848.000000	90848.000000	90848.000000	90848.000000	90848.000000	90848.000000
mean	46.551526	0.332533	87.808967	91.681809	0.776231	19.125914
std	21.759339	0.471124	15.848112	16.060918	0.416772	17.933869
min	17.000000	0.000000	35.000000	40.000000	0.000000	0.000000
25%	29.000000	0.000000	82.000000	81.000000	1.000000	7.000000
50%	44.000000	0.000000	95.000000	95.000000	1.000000	16.000000
75%	62.000000	1.000000	99.000000	106.000000	1.000000	25.000000
max	100.000000	1.000000	99.000000	110.000000	1.000000	76.000000

In [14]:

```
df_new.nunique()
```

Out[14]:

```
age         84
gender       2
spo2        65
pr          71
c/nc         2
oxy_flow    77
dtype: int64
```

In [15]:

```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [16]:

```
df_new['gender'].unique()
```

Out[16]:

```
array([0, 1], dtype=int64)
```

In [17]:

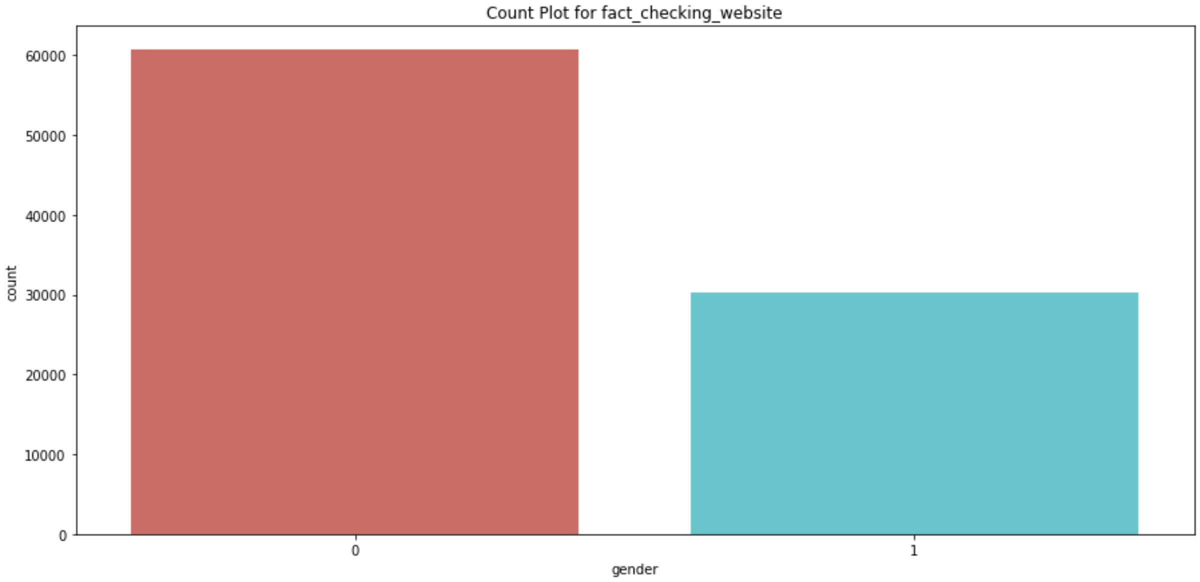
```
df_new['gender'].value_counts()
```

Out[17]:

```
0    60638
1    30210
Name: gender, dtype: int64
```

In [18]:

```
plt.figure(figsize=[15,7],)
plt.title('Count Plot for fact_checking_website')
sns.countplot(x = 'gender', data = df_new, palette = 'hls')
plt.show()
```



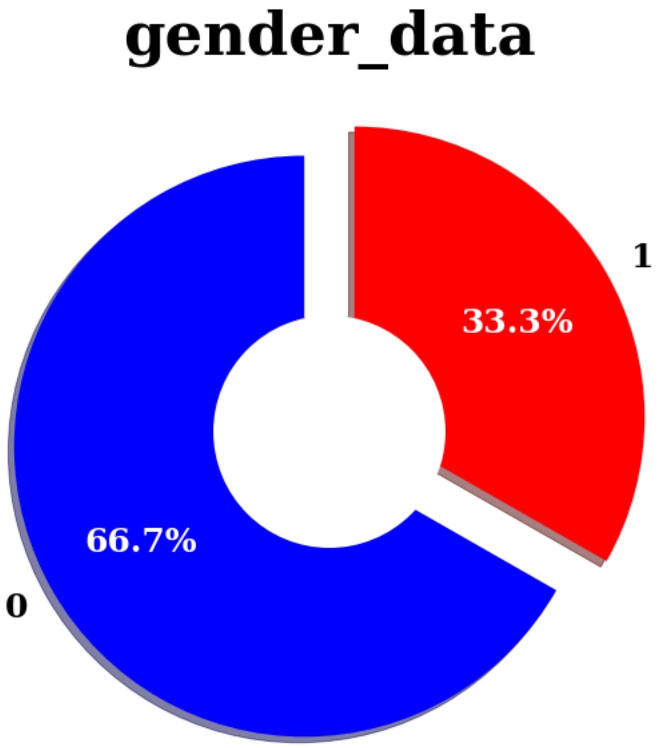
In [19]:

```
gender_data = df_new['gender'].value_counts()

explode = (0.1, 0.1)
plt.figure(figsize=(14, 10))
patches, texts, pcts = plt.pie(gender_data,
                               labels = gender_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 90,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 25,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='white')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('gender_data', size=45, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```



In [20]:

```
df_new['c/nc'].unique()
```

Out[20]:

```
array([1., 0.])
```

In [21]:

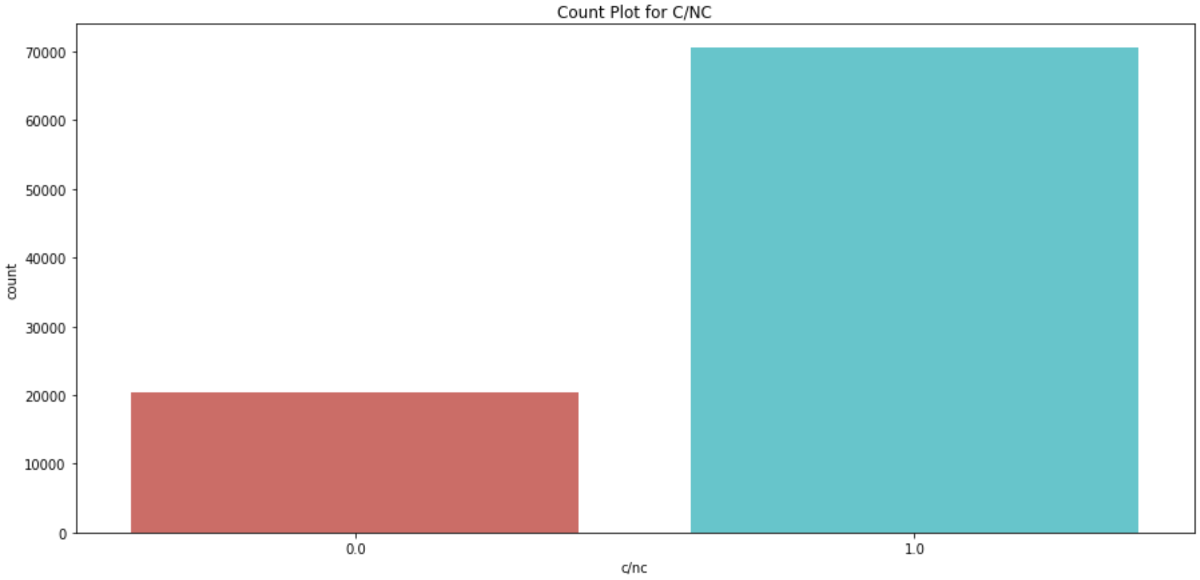
```
df_new['c/nc'].value_counts()
```

Out[21]:

```
1.0    70519
0.0    20329
Name: c/nc, dtype: int64
```

In [22]:

```
plt.figure(figsize=[15,7],)
plt.title('Count Plot for C/NC')
sns.countplot(x = 'c/nc', data = df_new, palette = 'hls')
plt.show()
```



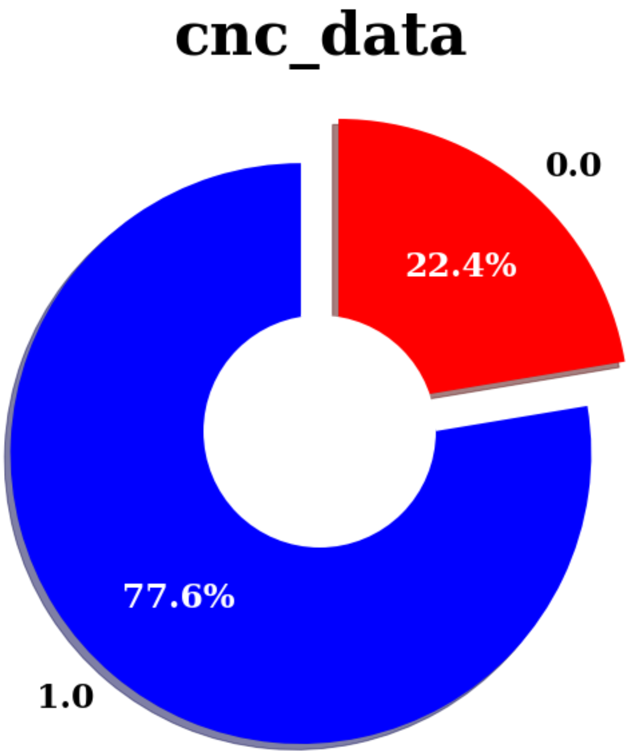
In [23]:

```
cnc_data = df_new['c/nc'].value_counts()

explode = (0.1, 0.1)
plt.figure(figsize=(14, 10))
patches, texts, pcts = plt.pie(cnc_data,
                               labels = cnc_data.index,
                               colors = ['blue', 'red'],
                               pctdistance = 0.65,
                               shadow = True,
                               startangle = 90,
                               explode = explode,
                               autopct = '%1.1f%%',
                               textprops={ 'fontsize': 25,
                                           'color': 'black',
                                           'weight': 'bold',
                                           'family': 'serif' })
plt.setp(pcts, color='white')

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('cnc_data', size=45, **hfont)

centre_circle = plt.Circle((0,0),0.40,fc='white')
fig = plt.gcf()
fig.gca().add_artist(centre_circle)
plt.show()
```

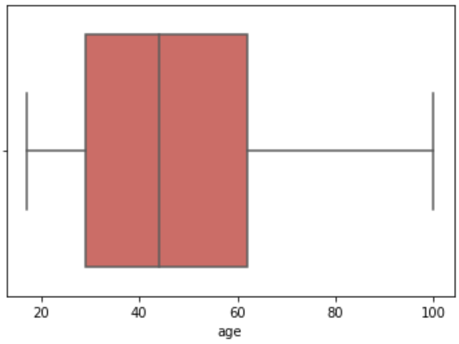


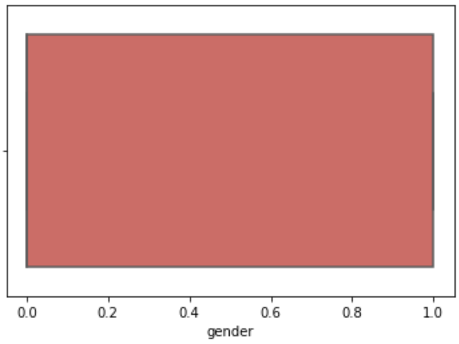
In [24]:

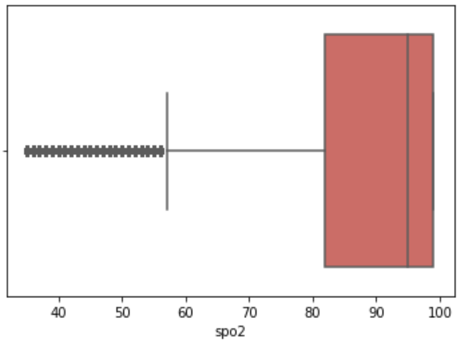
```
import warnings
warnings.filterwarnings('ignore')
```

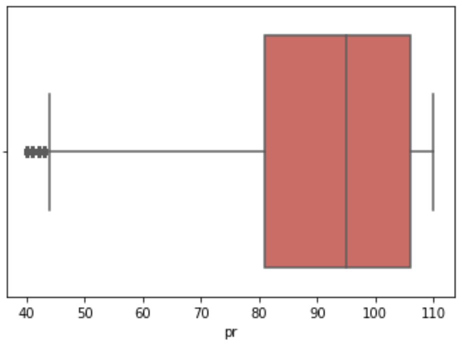
In [25]:

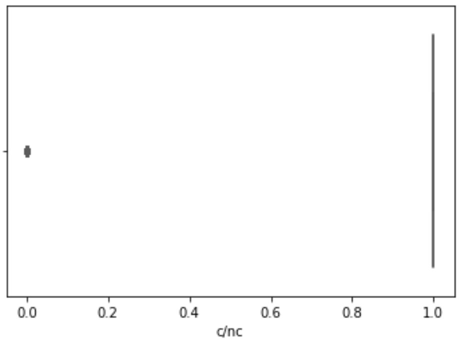
```
for i in df_new.columns:
    sns.boxplot(df_new[i], palette = 'hls')
    plt.show()
```

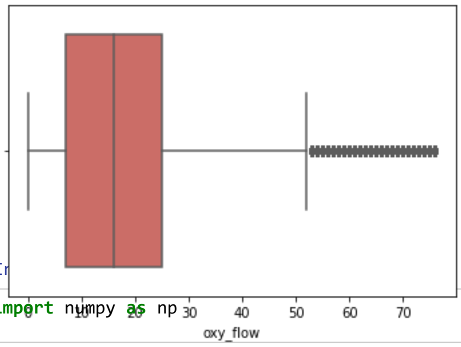












In

```
import numpy as np
```

In [27]:

```
from scipy import stats
df_new = df_new[(np.abs(stats.zscore(df_new)) < 3).all(axis=1)]
df_new.shape
```

Out[27]:

```
(83925, 6)
```

In [28]:

```
df_new.columns
```

Out[28]:

```
Index(['age', 'gender', 'spo2', 'pr', 'c/nc', 'oxy_flow'], dtype='object')
```

In [29]:

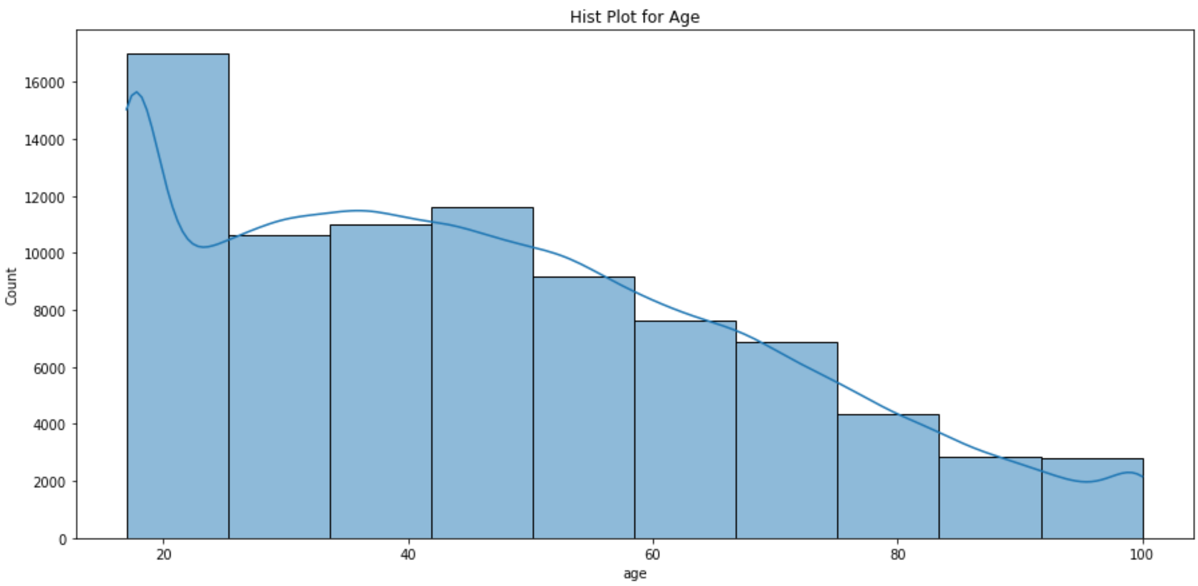
```
df_new.head()
```

Out[29]:

	age	gender	spo2	pr	c/nc	oxy_flow
0	27	0	74.0	72.0	1.0	6.0
4	52	0	69.0	84.0	1.0	0.0
5	82	0	93.0	93.0	1.0	28.0
9	68	0	90.0	92.0	1.0	33.0
13	40	0	99.0	109.0	1.0	27.0

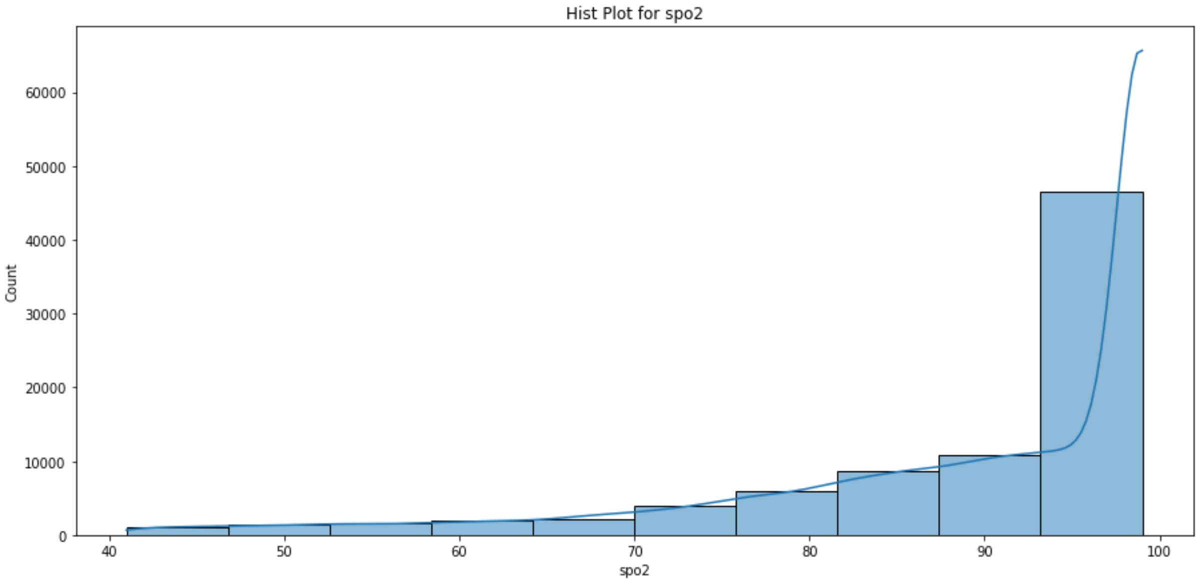
In [30]:

```
plt.figure(figsize=[15,7],)
plt.title('Hist Plot for Age')
sns.histplot(x = 'age', data = df_new, bins = 10, kde = True, palette = 'hls')
plt.show()
```



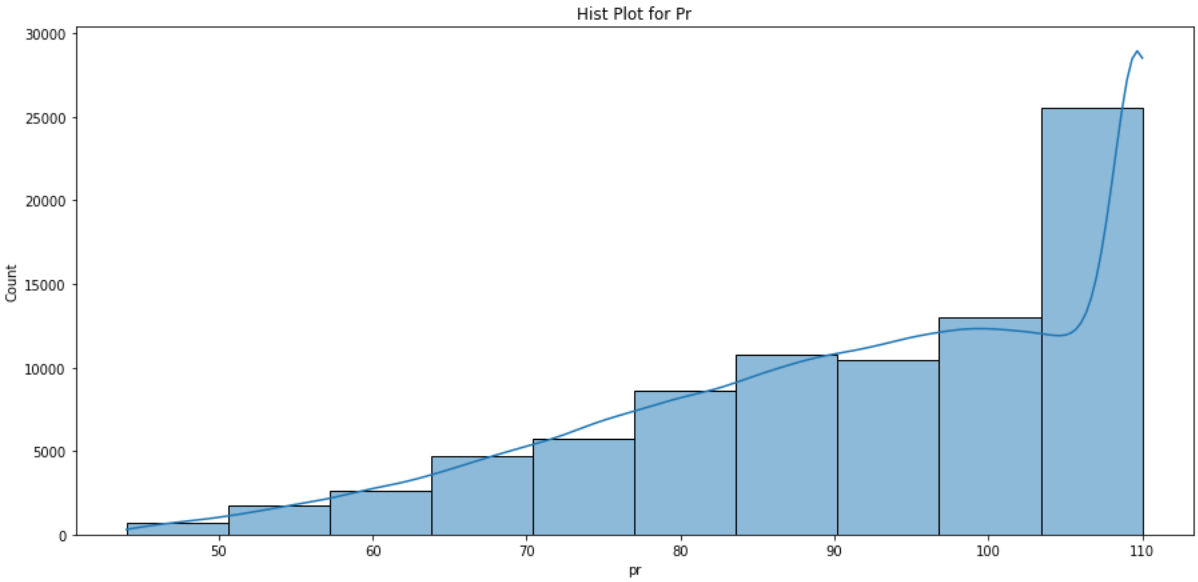
In [31]:

```
plt.figure(figsize=[15,7],)
plt.title('Hist Plot for spo2')
sns.histplot(x = 'spo2', data = df_new, bins = 10, kde = True, palette = 'hls')
plt.show()
```



In [32]:

```
plt.figure(figsize=[15,7],)
plt.title('Hist Plot for Pr')
sns.histplot(x = 'pr', data = df_new, bins = 10, kde = True, palette = 'hls')
plt.show()
```



In [33]:

```
df_corr = df_new.corr()
```

In [34]:

```python
plt.figure(figsize=[15,7],)
sns.heatmap(df_corr, annot = True)
plt.show()
```

In [35]:

```python
from sklearn.model_selection import train_test_split

X = df_new.iloc[:, :-1]
y = df_new.iloc[:, -1]
```

In [36]:

```python
from sklearn import preprocessing
```

In [37]:

```python
scaler = preprocessing.StandardScaler()
X = scaler.fit_transform(X)
```

In [38]:

```python
X_train, X_test, y_train, y_test = train_test_split(X, y, random_state=42, test_size=0.2)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[38]:

```
((67140, 5), (16785, 5), (67140,), (16785,))
```

In [39]:

```python
from sklearn.linear_model import LinearRegression

model_lr = LinearRegression()
model_lr.fit(X_train, y_train)
```

Out[39]:

```
▾ LinearRegression
LinearRegression()
```

In [40]:

```python
y_pred = model_lr.predict(X_test)
```

In [41]:

```python
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

In [42]:

```
print("MAE",mean_absolute_error(y_test,y_pred))
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
r2 = r2_score(y_test,y_pred)
print('R2 ',r2)
```

```
MAE 9.866725834475675
MSE 171.62334942211533
RMSE 13.100509510019652
R2  0.00935015491224922
```

In [43]:

```
import xgboost as xgb
xgb_r = xgb.XGBRegressor(objective ='reg:linear',
                         n_estimators = 10, seed = 123)
xgb_r.fit(X_train,y_train)
```

```
[14:47:07] WARNING: C:/Users/administrator/workspace/xgboost-win64_release_1.6.0/src/objective/regression_obj.cu:203: reg:l
inear is now deprecated in favor of reg:squarederror.
```

Out[43]:

```
XGBRegressor

XGBRegressor(base_score=0.5, booster='gbtree', callbacks=None,
             colsample_bylevel=1, colsample_bynode=1, colsample_bytree=1,
             early_stopping_rounds=None, enable_categorical=False,
             eval_metric=None, gamma=0, gpu_id=-1, grow_policy='depthwise',
             importance_type=None, interaction_constraints='',
             learning_rate=0.300000012, max_bin=256, max_cat_to_onehot=4,
             max_delta_step=0, max_depth=6, max_leaves=0, min_child_weight=1,
             missing=nan, monotone_constraints='()', n_estimators=10, n_jobs=0,
             num_parallel_tree=1, objective='reg:linear', predictor='auto',
             random_state=123, reg_alpha=0, ...)
```

In [44]:

```
y_pred = xgb_r.predict(X_test)
```

In [45]:

```
print("MAE",mean_absolute_error(y_test,y_pred))
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
r2 = r2_score(y_test,y_pred)
print('R2 ',r2)
```

```
MAE 9.862197978644986
MSE 172.36876321877892
RMSE 13.128928487076884
R2  0.005047453300390914
```

In [46]:

```
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor()
```

In [47]:

```
regressor.fit(X_train, y_train)
```

Out[47]:

```
DecisionTreeRegressor

DecisionTreeRegressor()
```

In [48]:

```
y_pred = regressor.predict(X_test)
```

In [49]:

```
print("MAE",mean_absolute_error(y_test,y_pred))
print("MSE",mean_squared_error(y_test,y_pred))
print("RMSE",np.sqrt(mean_squared_error(y_test,y_pred)))
r2 = r2_score(y_test,y_pred)
print('R2 ',r2)
```

```
MAE 13.397369255364445
MSE 317.6290358851169
RMSE 17.822150147642592
R2  -0.8334285879775272
```

In [50]:

```
#Conclusion: The data or the models used are not sufficient to predict the
# oxygen flow for a patient.
```