```
#Working with Simple Linear Regression
#Risk analysis of space shuttle
```

```
import os
os.chdir("C:/Users/Mohan/Desktop/Datasets")
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

```
challenger=pd.read_csv('challenger.csv')
```

```
challenger
```

	o_ring_ct	O.ring.failures	temperature	pressure	launch_id
0	6	0	66	50	1
1	6	1	70	50	2

```
#Assign ring failure values to an object
O_ring_failures=challenger['O.ring.failures']
```

```
O_ring_failures
```

```
0     0
1     1
2     0
3     0
4     0
5     0
6     0
7     0
8     1
9     1
10    1
11    0
12    0
13    2
14    0
15    0
16    0
17    0
18    0
19    0
20    0
21    0
22    1
Name: O.ring.failures, dtype: int64
```

19	6	0	79	200	20

```
#Assign temperature values to 'temp' object


temp=challenger['temperature']
```

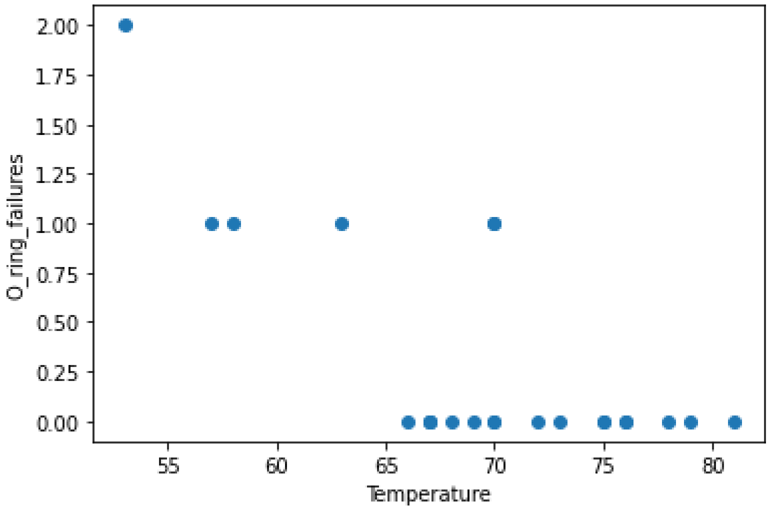
```
temp
```

```
0     66
1     70
2     69
3     68
4     67
5     72
6     73
7     70
8     57
9     63
10    70
11    78
12    67
13    53
```

```
14    67
15    75
16    70
17    81
18    76
19    79
20    75
21    76
22    58
Name: temperature, dtype: int64
```

```
import matplotlib.pyplot as plt
%matplotlib inline
plt.plot(temp,O_ring_failures,'o')
plt.ylabel("O_ring_failures")
plt.xlabel("Temperature")
```

```
Text(0.5, 0, 'Temperature')
```




```
challenger.corr()
```

	o_ring_ct	O.ring.failures	temperature	pressure	launch_id
o_ring_ct	NaN	NaN	NaN	NaN	NaN
O.ring.failures	NaN	1.000000	-0.725671	0.220326	-0.011993
temperature	NaN	-0.725671	1.000000	0.039818	0.230770
pressure	NaN	0.220326	0.039818	1.000000	0.839932
launch_id	NaN	-0.011993	0.230770	0.839932	1.000000

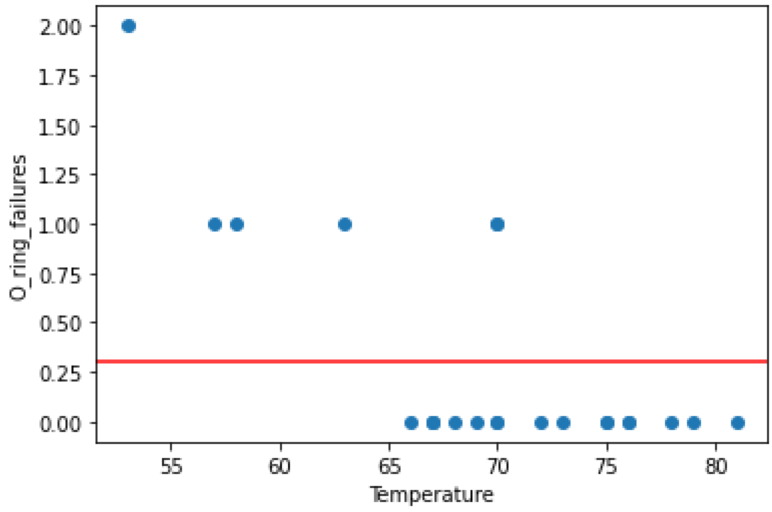
```
mean_O_ring_failures = challenger['O.ring.failures'].mean()
```

```
mean_O_ring_failures
```

```
0.30434782608695654
```

```
plt.plot(temp,O_ring_failures,'o')
```

```
plt.ylabel("O_ring_failures")
plt.xlabel("Temperature")
plt.axhline(mean_O_ring_failures, color='r', linestyle='-')
plt.show()
```




```
import statsmodels.api as sm
model=sm.OLS(O_ring_failures,temp).fit()
```

```
#Obtain model summary
model.summary()
```

OLS Regression Results

Dep. Variable:	O.ring.failures	**R-squared (uncentered):**	0.178
Model:	OLS	**Adj. R-squared (uncentered):**	0.140
Method:	Least Squares	**F-statistic:**	4.755
Date:	Mon, 10 Oct 2022	**Prob (F-statistic):**	0.0402
Time:	06:24:21	**Log-Likelihood:**	-19.595
No. Observations:	23	**AIC:**	41.19
Df Residuals:	22	**BIC:**	42.33
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
temperature	0.0038	0.002	2.181	0.040	0.000	0.007

Omnibus:	14.053	**Durbin-Watson:**	1.851
Prob(Omnibus):	0.001	**Jarque-Bera (JB):**	12.853
Skew:	1.623	**Prob(JB):**	0.00162
Kurtosis:	4.696	**Cond. No.**	1.00

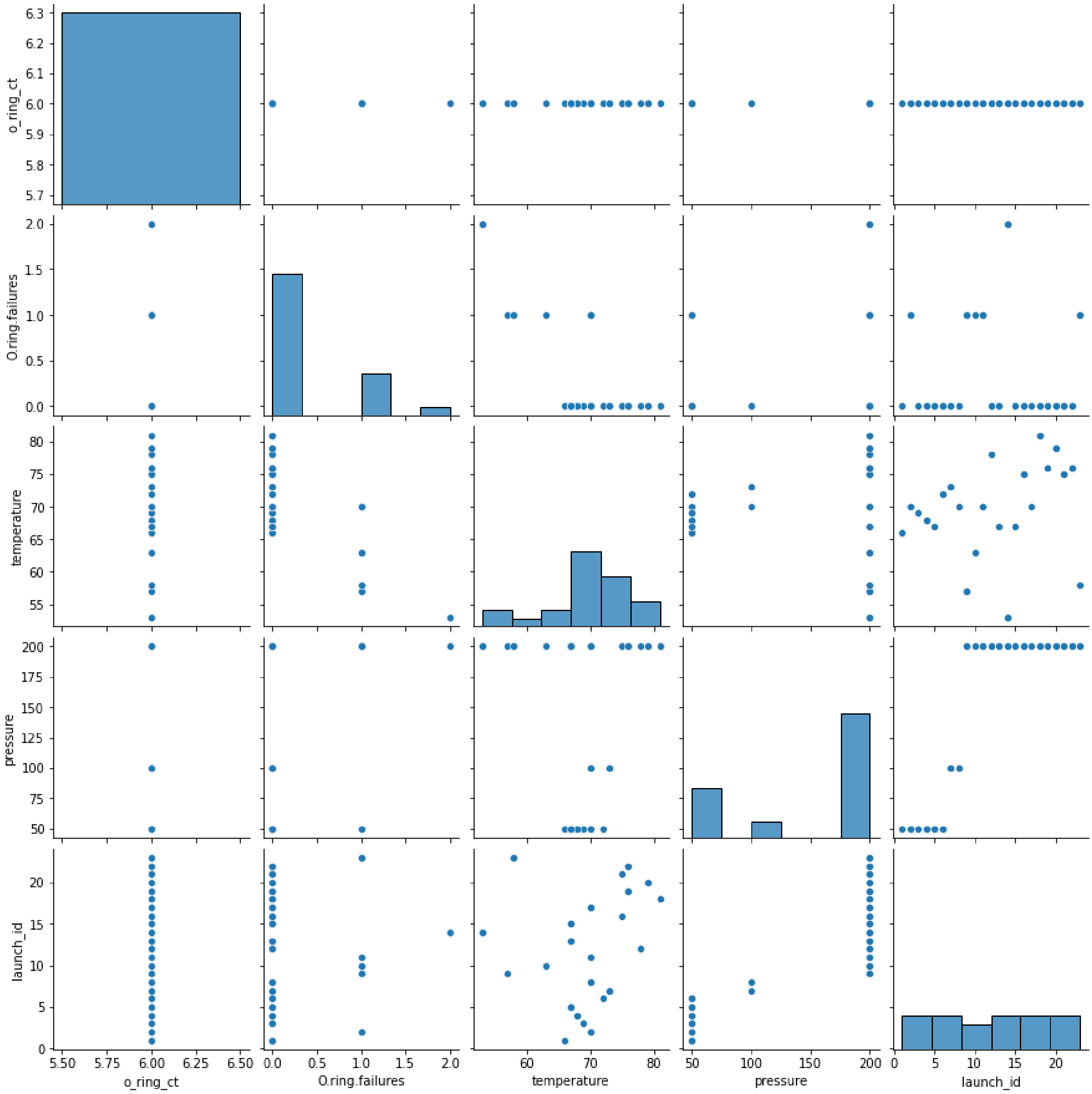
Notes:
[1] R^2 is computed without centering (uncentered) since the model does not contain a constant.
[2] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Observation

As temeparture increases by 1 degree, O.ring.failures increase by 0.0038

```
import seaborn as sns
sns.pairplot(challenger)
```

```
<seaborn.axisgrid.PairGrid at 0x7fcdd3df6110>
```

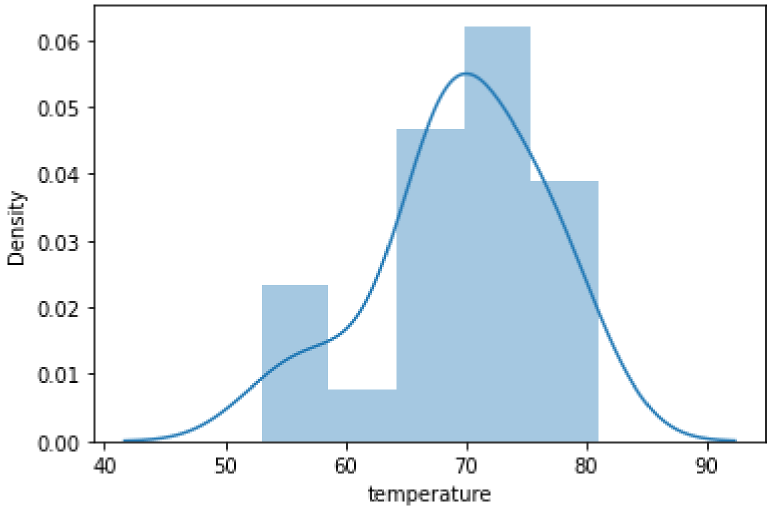


Observations

1.The histogram on the diagonal allows us to see the distribution of a single variable 2.The scatter plots on the upper and lower triangles show the relationship (or lack thereof) between two variables.

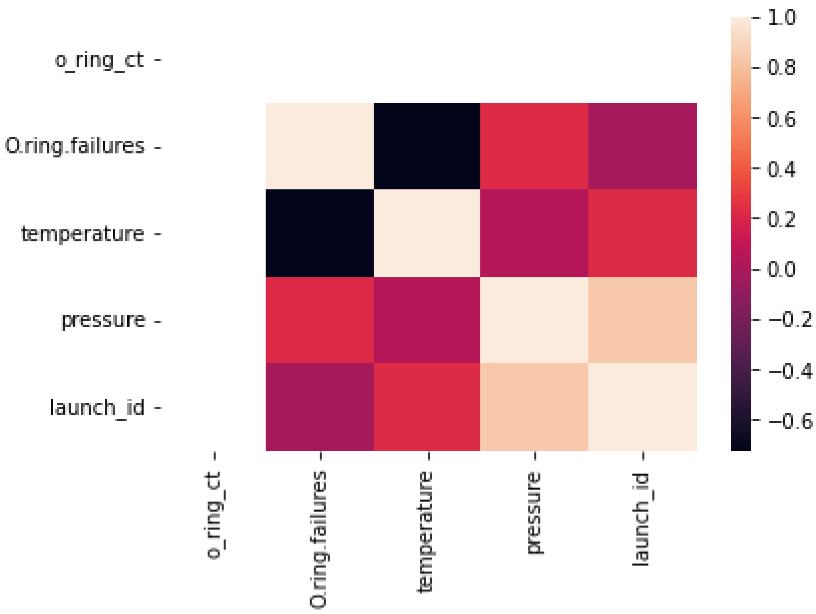
```
sns.distplot(challenger['temperature'])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning:
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7fcdd3873690>
```




```
# Plot the correlation using heatmap
corr = challenger.corr()
sns.heatmap(corr,xticklabels=corr.columns,yticklabels=corr.columns)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fcdd380aed0>
```



Observations

Black color represents negative correlation which exists betewen temeprature and O.ring.failures

[ ] ↳ *12 cells hidden*

Linear Regression with sklearn

```
from sklearn import linear_model as lm
model=lm.LinearRegression()
results=model.fit(X_train,y_train)
```

```
accuracy = model.score(X_train, y_train)
print('Accuracy of the model:', accuracy)
```

```
Accuracy of the model: 0.496692522077835
```

```
#Print coefficients
print('intercept:', model.intercept_)
print('slope:', model.coef_)
```

```
intercept: 3.874282260501661
slope: [-0.05197945]
```

Observation

The slope value -0.051 means that the predicted O.ring.failures reduces by -0.05 when x(temperature) rises by one degree

```
#Check model score
model
```

```
LinearRegression()
```

```
X_test
```

	temperature	pressure
20	75	200
17	81	200
3	68	50
13	53	200
19	79	200
4	67	50

```
X_test.shape
```

```
(20, 1)
```

```
y_test.shape
```

```
(10,)
```

```
X_test = X_test.values.reshape((-1,1))
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-97-6fc1b52b26db> in <module>
----> 1 X_test = X_test.values.reshape((-1,1))

AttributeError: 'numpy.ndarray' object has no attribute 'values'
```

SEARCH STACK OVERFLOW

```
#Predictions from the model
predictions = model.predict(X_test)
print('predicted O.ring.failures:',predictions, sep = '\n')
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-95-111a443fb772> in <module>
      1 #Predictions from the model
----> 2 predictions = model.predict(X_test)
      3 print('predicted O.ring.failures:',predictions, sep = '\n')
```

3 frames

```
/usr/local/lib/python3.7/dist-packages/sklearn/base.py in _check_n_features(self, X, reset)
    399         if n_features != self.n_features_in_:
    400             raise ValueError(
--> 401                 f"X has {n_features} features, but {self.__class__.__name__}
"
    402                 f"is expecting {self.n_features_in_} features as input."
    403             )

ValueError: X has 1 features, but LinearRegression is expecting 2 features as input.
```

```
y_test
```

```
#Visualize the predictions
plt.scatter(y_test, predictions)
```

Observations

A linear model has been obtained

```
#Other way for prediction
y_pred = model.intercept_ + model.coef_ * X_test
print('predicted response:', y_pred, sep='\n')
```

```
predicted response:
[[-0.02417649]
 [-0.33605319]
 [ 0.33967966]
 [ 1.11937141]
 [-0.23209429]
 [ 0.23572076]
 [ 0.23572076]
 [ 0.39165911]
 [ 0.28770021]
 [-0.07615594]]
```

```
#Define new data instance
Xnew = [[30]]

#Make a Prediction
ynew = model.predict(Xnew)

#Show the inputs and predicted outputs
print("New Temperature=%s, Predicted O.ring.failures=%s" % (Xnew,ynew))
```

```
New Temperature=[[30]], Predicted O.ring.failures=[2.31489876]
```

```
#Define new data instance
Xnew = [[70]]

#Make a Prediction
ynew = model.predict(Xnew)

#Show the inputs and predicted outputs
print("New Temperature=%s, Predicted O.ring.failures=%s" % (Xnew,ynew))
```

```
New Temperature=[[70]], Predicted O.ring.failures=[0.23572076]
```

```
#Evaluating the model
from sklearn.metrics import mean_squared_error, r2_score
X_train = X_train.reshape(-1,1)
y_train_prediction = model.predict(X_train)
```

```
X_test = X_test.reshape(-1,1)
y_test_prediction = model.predict(X_test)


# printing values
print('Slope:' ,model.coef_)
print('Intercept:', model.intercept_)
print("\n")

# model evaluation for training set
import numpy as np
rmse_training = (np.sqrt(mean_squared_error(y_train, y_train_prediction)))
r2_training = r2_score(y_train, y_train_prediction)

print("The model performance for training set")
print("--------------------------------------")
print('RMSE is {}'.format(rmse_training))
print('R2 score is {}'.format(r2_training))
print("\n")

# model evaluation for testing set
rmse_testing = (np.sqrt(mean_squared_error(y_test, y_test_prediction)))
r2_testing = r2_score(y_test, y_test_prediction)

print("The model performance for testing set")
print("--------------------------------------")
print('Root mean squared error: ', rmse_testing)
print('R2 score: ', r2_testing)
```

```
Slope: [-0.05197945]
Intercept: 3.874282260501661


The model performance for training set
--------------------------------------
RMSE is 0.32743461522828027
R2 score is 0.496692522077835


The model performance for testing set
--------------------------------------
Root mean squared error:  0.4404461397642911
R2 score:  0.5268468243091087
```

```
# plotting values

# data points
plt.scatter(X, y)
plt.xlabel('Temeperature')
plt.ylabel('O.ring.failures')
```

```
Text(0, 0.5, 'O.ring.failures')
```


```
X= X.reshape(-1,1)
y_predicted = model.predict(X)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-70-4f01f4a3466f> in <module>
----> 1 X= X.reshape(-1,1)
      2 y_predicted = model.predict(X)

/usr/local/lib/python3.7/dist-packages/pandas/core/generic.py in __getattr__(self,
name)
   5485         ):
   5486             return self[name]
-> 5487         return object.__getattribute__(self, name)
   5488 
   5489     def __setattr__(self, name: str, value) -> None:

AttributeError: 'Series' object has no attribute 'reshape'
```

SEARCH STACK OVERFLOW

```
# predicted values
plt.plot(X, y_predicted, color='r')
plt.show()
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-89-839323038b49> in <module>
      1 # predicted values
----> 2 plt.plot(X, y_predicted, color='r')
      3 plt.show()

NameError: name 'y_predicted' is not defined
```

SEARCH STACK OVERFLOW

Multiple Linear Regression

[ ] ↳ *2 cells hidden*

Observation

1.This model has a higher R-squared compared to simple linear model against temperature and O.ring.failure 2.However in this model both temeprature and pressure features became statistcially insignificant to predict O.ring.failure 3.As pressure increases by 1 atmosphere, O.ring.failures increase by 0.0031 and as temperature increases by 1 degree, O.ring.failures decrease by -0.0030

[ ] ↳ *12 cells hidden*

Observation

With both temeprature and pressure the model is not linear

[ ] ↳ *2 cells hidden*