```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder,StandardScaler
from sklearn.linear_model import LinearRegression,Lasso
from sklearn.metrics import mean_squared_error,mean_absolute_error
from sklearn.ensemble import RandomForestRegressor
import warnings
warnings.filterwarnings("ignore")

Salary=pd.read_csv(r'C:\Users\21F22227\Desktop\Salary.csv')
Salary.head()
Salary.shape
Salary.isnull().sum()
Salary=Salary.dropna()
Salary.isnull().sum()
Salary.shape
Salary.dtypes

plt.figure(figsize=(10,6))
sns.regplot(x=" Experience", y="Salary", data=Salary)

from scipy import stats
pearson_coef, p_value = stats.pearsonr(Salary['Age'],Salary['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

plt.figure(figsize=(10,6))
sns.regplot(x=" Experience", y="Age", data=Salary)
```

```
from scipy import stats
pearson_coef, p_value = stats.pearsonr(Salary['Age'],Salary['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)

plt.figure(figsize=(10,6))
sns.boxplot(x="Gender", y="Salary", data=Salary)

plt.figure(figsize=(10,6))
sns.boxplot(x="Dgree", y="Salary", data=Salary)

plt.figure(figsize=(10,6))
sns.boxplot(x="Position", y="Salary", data=Salary)

Salary.shape
Salary.describe()
Salary.describe(include=['object'])

from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
Salary.Gender= labelencoder.fit_transform(Salary.Gender)
Salary.Dgree= labelencoder.fit_transform(Salary.Dgree)
Salary.Position= labelencoder.fit_transform(Salary.Position)

Salary.head(10)

import scipy.stats as stats
Salary = stats.zscore(Salary)
Salary = stats.zscore(Salary)
```

```
Salary
x_train=Salary.iloc[:,0:4]
y_train=Salary.iloc[:,4]
x_test=Salary.iloc[:,0:4]
y_test=Salary.iloc[:,4]

x_train

rg = LinearRegression()
mdl=rg.fit(x_train,y_train)

y_pred1 = rg.predict(x_test)

print('The R-square for Multiple Linear regression is: ',
rg.score(x_train,y_train))

mse1 = mean_squared_error(y_test, y_pred1)
print('The mean square error for Multiple Linear Regression: ', mse1)

mae1= mean_absolute_error(y_test, y_pred1)
print('The mean absolute error for Multiple Linear Regression: ', mae1)

plt.figure(figsize=(10,6))
ax1 = sns.distplot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(y_pred1, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for Price')
plt.xlabel('Price (in dollars)')
plt.ylabel('Proportion of Cars')
plt.show()
```

```
plt.close()

rf = RandomForestRegressor()
model=rf.fit(x_train,y_train)

y_pred2 = rf.predict(x_test)
print('The R-square for Random Forest is: ', rf.score(x_train,y_train))
mse2 = mean_squared_error(y_test, y_pred2)
print('The mean square error of price and predicted value is: ', mse2)
mae2= mean_absolute_error(y_test, y_pred2)
print('The mean absolute error of price and predicted value is: ', mae2)

plt.figure(figsize=(10,6))
ax1 = sns.distplot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(y_pred2, hist=False, color="b", label="Fitted Values" , ax=ax1)
plt.title('Actual vs Fitted Values for Price')
plt.xlabel('Price (in dollars)')
plt.ylabel('Proportion of Cars')
plt.show()
plt.close()

LassoModel=Lasso()
lm=LassoModel.fit(x_train,y_train)

y_pred3 = lm.predict(x_test)
print('The R-square for LASSO is: ', lm.score(x_train,y_train))

mae3= mean_absolute_error(y_test, y_pred3)
print('The mean absolute error of price and predicted value is: ', mae3)
```

```
mse3 = mean_squared_error(y_test, y_pred3)
print('The mean square error of price and predicted value is: ', mse3)

scores = [('MLR', mae1),
 ('Random Forest', mae2),
 ('LASSO', mae3) ]

mae = pd.DataFrame(data = scores, columns=['Model', 'MAE Score'])
mae

mae.sort_values(by=(['MAE Score']), ascending=False, inplace=True)
f, axe = plt.subplots(1,1, figsize=(10,7))
sns.barplot(x = mae['Model'], y=mae['MAE Score'], ax = axe)
axe.set_xlabel('Mean Absolute Error', size=20)
axe.set_ylabel('Model', size=20)
plt.show()
```