In [1]:
```
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")
```

In [2]:
```
data = pd.read_csv(r'D:\data\Engineering_graduate_salary.csv')
data.head()
```

Out[2]:

	Gender	percentage	board	graduation	percentage.1	CollegeTier	Specialization	collegeGPA	CollegeCityID	CollegeCityTier	...	Logical	Quant	I
0	f	87.80	cbse	2009	84.00	1	instrumentation and control engineering	73.82	6920	1	...	665.0	810	0.
1	m	57.00	cbse	2010	64.50	2	computer science & engineering	65.00	6624	0	...	435.0	210	0.
2	m	77.33	maharashtra state board,pune	2007	85.17	2	electronics & telecommunications	61.94	9084	0	...	475.0	505	0.
3	m	84.30	cbse	2009	86.00	1	computer science & engineering	80.40	8195	1	...	NaN	635	0.
4	f	82.00	cbse	2008	75.00	2	biotechnology	64.30	4889	1	...	495.0	365	0.

5 rows × 22 columns

In [3]:
```
data.shape
```

Out[3]:
```
(2998, 22)
```

In [4]:
```
data.isnull().sum()
```

Out[4]:
```
Gender                    0
percentage                0
board                     0
graduation                0
percentage.1              0
CollegeTier               0
Specialization            0
collegeGPA                0
CollegeCityID             0
CollegeCityTier           0
CollegeState              0
English                   0
Logical                  11
Quant                     0
Domain                    0
ComputerProgramming       0
conscientiousness         0
agreeableness             0
extraversion              0
nueroticism               0
openess_to_experience     0
Salary                    0
dtype: int64
```

In [5]:
```
data=data.dropna()
```

In [6]:
```
data.isnull().sum()
```

Out[6]:
```
Gender                   0
percentage               0
board                    0
graduation               0
percentage.1             0
CollegeTier              0
Specialization           0
collegeGPA               0
CollegeCityID            0
CollegeCityTier          0
CollegeState             0
English                  0
Logical                  0
Quant                    0
Domain                   0
ComputerProgramming      0
conscientiousness        0
agreeableness            0
extraversion             0
nueroticism              0
openess_to_experience    0
Salary                   0
dtype: int64
```

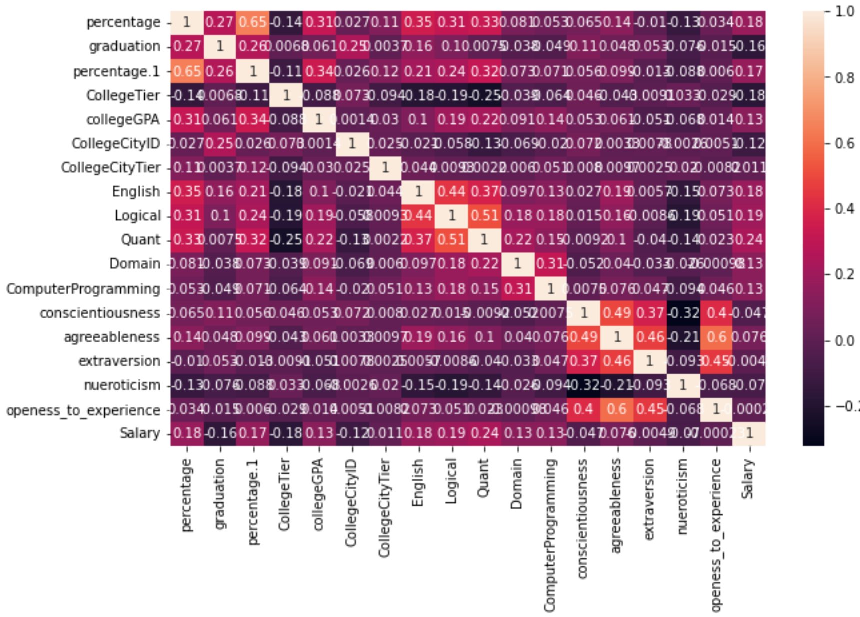
In [7]:
```
data.shape
```

Out[7]:
```
(2987, 22)
```

In [8]:
```
data.dtypes
```

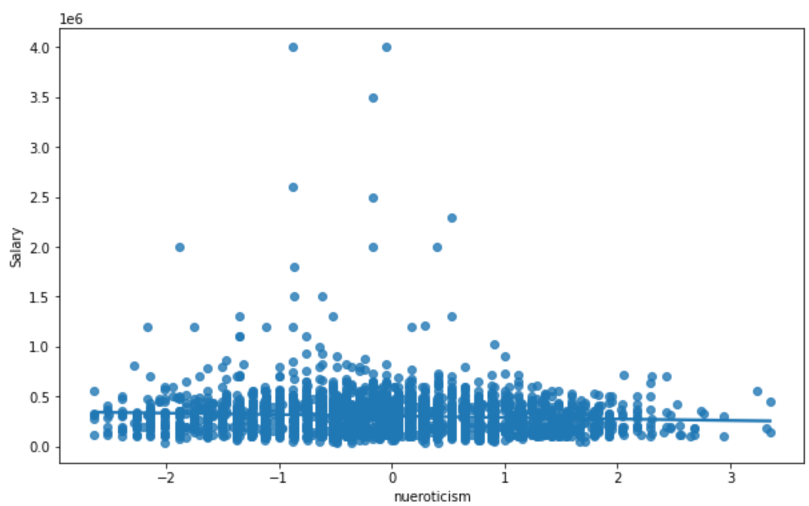
Out[8]:
```
Gender                    object
percentage               float64
board                     object
graduation                 int64
percentage.1             float64
CollegeTier                int64
Specialization            object
collegeGPA               float64
CollegeCityID              int64
CollegeCityTier            int64
CollegeState              object
English                    int64
Logical                  float64
Quant                      int64
Domain                   float64
ComputerProgramming        int64
conscientiousness        float64
agreeableness            float64
extraversion             float64
nueroticism              float64
openess_to_experience    float64
Salary                     int64
dtype: object
```

In [9]:
```
plt.figure(figsize=(10,6))
corr = data.corr()
sns.heatmap(corr,annot=True)
plt.show()
```



In [10]:
```
plt.figure(figsize=(10,6))
sns.regplot(x="nueroticism", y="Salary", data=data)
```

Out[10]: `<AxesSubplot:xlabel='nueroticism', ylabel='Salary'>`

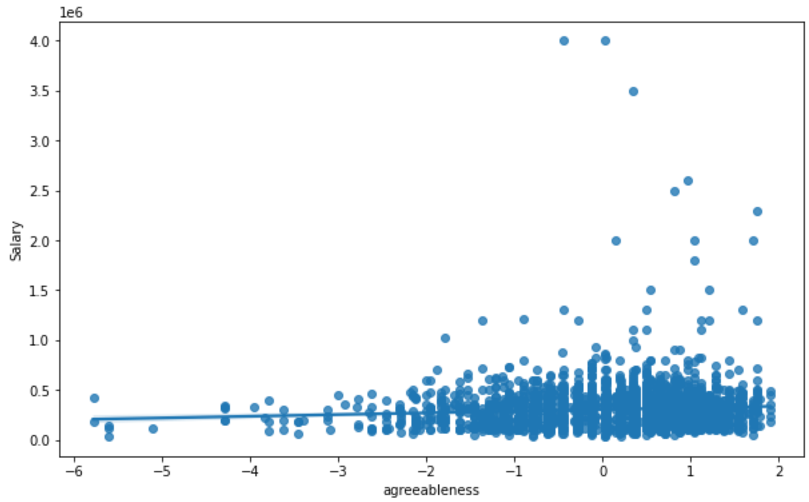


In [11]:
```
from scipy import stats
pearson_coef, p_value = stats.pearsonr(data['nueroticism'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.07018342952371795  with a P-value of P = 0.00012358897278919818
```

In [12]:
```
plt.figure(figsize=(10,6))
sns.regplot(x="agreeableness", y="Salary", data=data)
```

Out[12]: <AxesSubplot:xlabel='agreeableness', ylabel='Salary'>

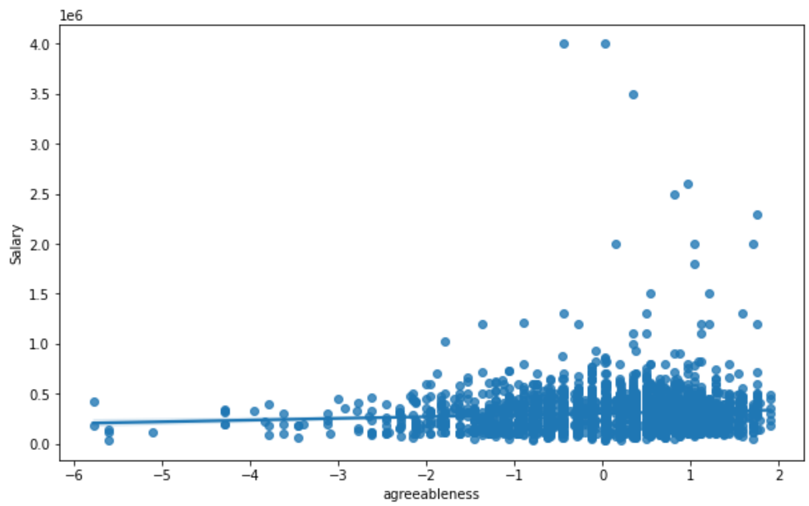


In [13]:
```
pearson_coef, p_value = stats.pearsonr(data['agreeableness'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

The Pearson Correlation Coefficient is 0.07561337581760656 with a P-value of P = 3.522974623965606e-05

In [14]:
```
plt.figure(figsize=(10,6))
sns.regplot(x="agreeableness", y="Salary", data=data)
```

Out[14]: <AxesSubplot:xlabel='agreeableness', ylabel='Salary'>

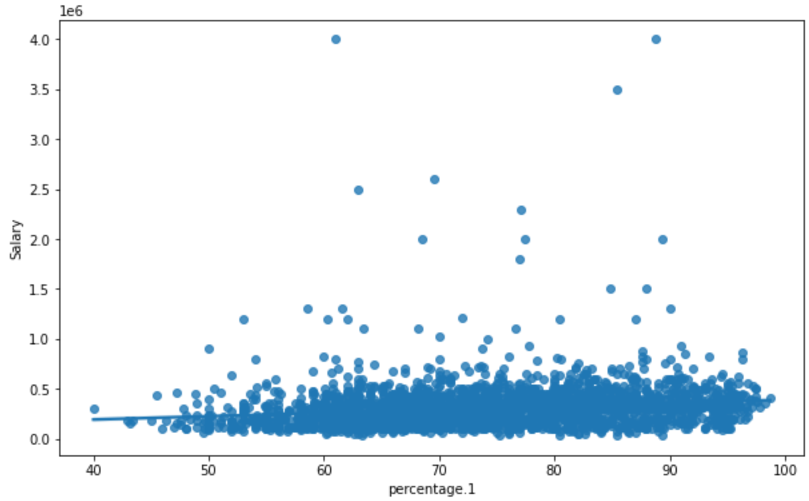


In [15]:
```
pearson_coef, p_value = stats.pearsonr(data['agreeableness'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

The Pearson Correlation Coefficient is 0.07561337581760656 with a P-value of P = 3.522974623965606e-05

In [16]:
```
plt.figure(figsize=(10,6))
sns.regplot(x="percentage.1", y="Salary", data=data)
```

Out[16]: <AxesSubplot:xlabel='percentage.1', ylabel='Salary'>

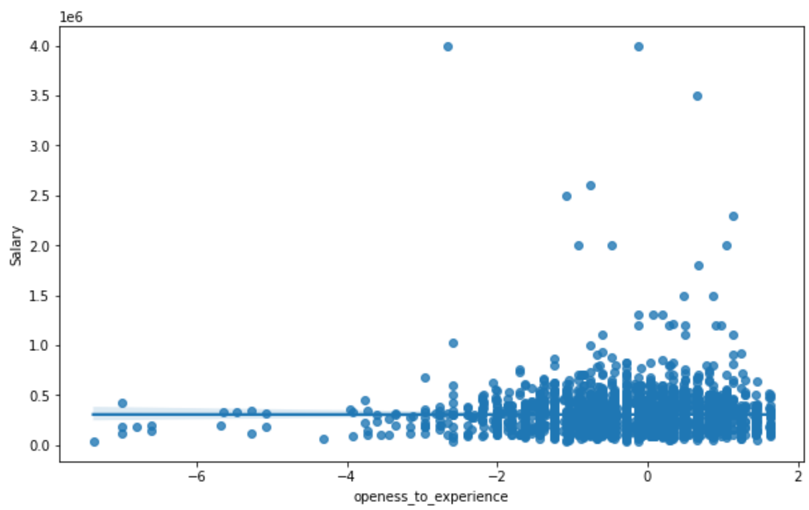


In [17]:
```
pearson_coef, p_value = stats.pearsonr(data['percentage.1'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is 0.17231615705502723  with a P-value of P = 2.429408205521456e-21
```

In [18]:
```
plt.figure(figsize=(10,6))
sns.regplot(x="openess_to_experience", y="Salary", data=data)
```

Out[18]: <AxesSubplot:xlabel='openess_to_experience', ylabel='Salary'>

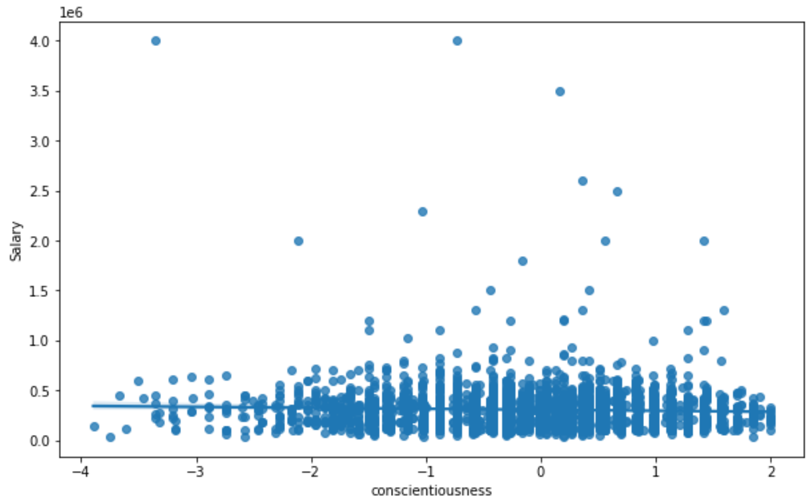


In [19]:
```
pearson_coef, p_value = stats.pearsonr(data['openess_to_experience'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.0002265485013701954  with a P-value of P = 0.9901252532076682
```

In [20]:
```python
plt.figure(figsize=(10,6))
sns.regplot(x="conscientiousness", y="Salary", data=data)
```

Out[20]:
```
<AxesSubplot:xlabel='conscientiousness', ylabel='Salary'>
```

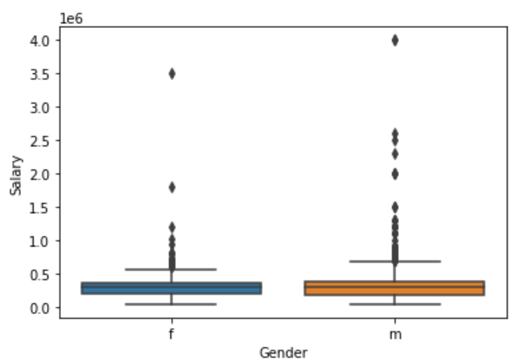


In [21]:
```python
pearson_coef, p_value = stats.pearsonr(data['conscientiousness'], data['Salary'])
print("The Pearson Correlation Coefficient is", pearson_coef, " with a P-value of P =", p_value)
```

```
The Pearson Correlation Coefficient is -0.04691712946297745  with a P-value of P = 0.010331799649872648
```

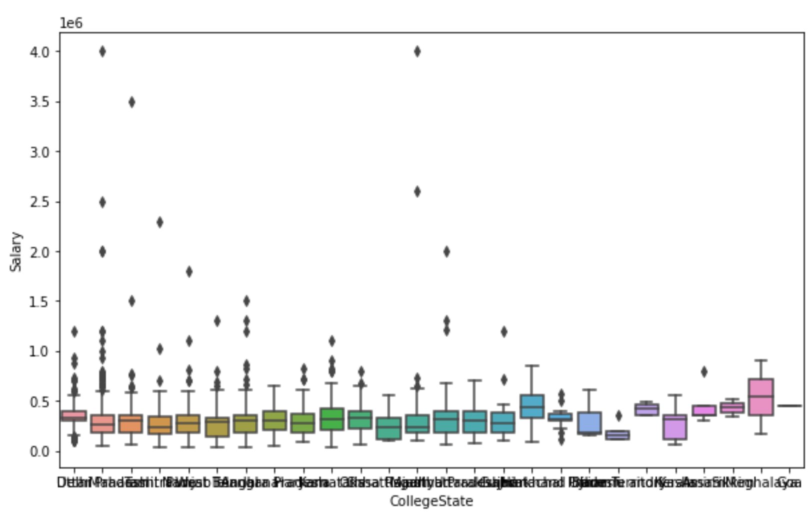
In [22]:
```python
sns.boxplot(x="Gender", y="Salary", data=data)
```

Out[22]:
```
<AxesSubplot:xlabel='Gender', ylabel='Salary'>
```



In [23]:
```python
plt.figure(figsize=(10,6))
sns.boxplot(x="CollegeState", y="Salary", data=data)
```

Out[23]:
```
<AxesSubplot:xlabel='CollegeState', ylabel='Salary'>
```



In [24]:
```
data.drop(['Logical', 'conscientiousness', 'extraversion','graduation'], axis = 1, inplace = True)
```

In [25]:
```
data.shape
```

Out[25]:
```
(2987, 18)
```

In [26]:
```
data.describe()
```

Out[26]:

	percentage	percentage.1	CollegeTier	collegeGPA	CollegeCityID	CollegeCityTier	English	Quant	Domain	ComputerProgramming	agre
count	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	2987.000000	29
mean	77.692444	74.349026	1.925008	71.495795	5210.343488	0.296284	501.199197	514.213592	0.507850	351.787412	
std	9.980721	11.120562	0.263422	8.127308	4778.230951	0.456694	105.262871	122.187258	0.463853	204.559332	
min	43.000000	40.000000	1.000000	6.630000	2.000000	0.000000	180.000000	120.000000	-1.000000	-1.000000	
25%	71.200000	66.000000	2.000000	66.500000	526.000000	0.000000	425.000000	430.000000	0.342315	295.000000	
50%	79.000000	74.000000	2.000000	71.800000	4032.000000	0.000000	500.000000	515.000000	0.622643	415.000000	
75%	85.600000	82.600000	2.000000	76.300000	8823.000000	1.000000	570.000000	595.000000	0.833603	495.000000	
max	97.760000	98.700000	2.000000	99.930000	18409.000000	1.000000	875.000000	900.000000	0.999910	804.000000	

In [27]:
```
data['Salary']
```

Out[27]:
```
0       445000
1       110000
2       255000
4       200000
5       440000
         ...
2993    120000
2994    120000
2995    385000
2996    530000
2997    200000
Name: Salary, Length: 2987, dtype: int64
```

In [28]:
```
data.describe(include=['object'])
```

Out[28]:

	Gender	board	Specialization	CollegeState
count	2987	2987	2987	2987
unique	2	218	42	26
top	m	cbse	electronics and communication engineering	Uttar Pradesh
freq	2273	1024	669	695

In [29]:
```
from sklearn.preprocessing import LabelEncoder
labelencoder = LabelEncoder()
data.Gender = labelencoder.fit_transform(data.Gender)
data.board = labelencoder.fit_transform(data.board)
data.Specialization = labelencoder.fit_transform(data.Specialization)
data.CollegeState = labelencoder.fit_transform(data.CollegeState)
```

In [30]:
```
data.head(10)
```

Out[30]:

	Gender	percentage	board	percentage.1	CollegeTier	Specialization	collegeGPA	CollegeCityID	CollegeCityTier	CollegeState	English	Quant	Domain	C
0	0	87.80	45	84.00	1	33	73.82	6920	1	4	650	810	0.694479	
1	1	57.00	45	64.50	2	12	65.00	6624	0	23	440	210	0.342315	
2	1	77.33	126	85.17	2	19	61.94	9084	0	14	485	505	0.824666	
4	0	82.00	45	75.00	2	4	64.30	4889	1	20	575	365	0.278457	
5	0	83.16	72	77.00	1	33	99.93	10950	0	17	535	620	0.376060	
6	0	72.50	177	53.20	2	37	68.00	14381	1	25	510	405	0.829585	
7	0	77.00	177	88.00	2	12	71.00	13208	1	21	370	280	0.704090	
8	1	76.80	177	87.70	2	32	73.15	5338	0	0	510	440	0.744758	
10	1	77.00	177	75.00	2	19	62.00	13424	0	14	675	485	0.207392	
11	1	81.20	177	79.90	2	33	67.67	64	0	23	395	645	-1.000000	

In [31]:
```
import scipy.stats as stats
data = stats.zscore(data)
data = stats.zscore(data)
```

In [32]:
```
data
```

Out[32]:

	Gender	percentage	board	percentage.1	CollegeTier	Specialization	collegeGPA	CollegeCityID	CollegeCityTier	CollegeState	English	Quant
0	-1.784229	1.012878	-0.813124	0.867995	-3.512096	1.495589	0.286023	0.357861	1.541149	-1.723469	1.413848	2.421168
1	0.560466	-2.073588	-0.813124	-0.885807	0.284730	-0.812254	-0.799389	0.295903	-0.648866	0.933600	-0.581491	-2.490149
2	0.560466	-0.036320	0.369883	0.973223	0.284730	-0.042973	-1.175961	0.810824	-0.648866	-0.325011	-0.153919	-0.075418
4	-1.784229	0.431660	-0.813124	0.058548	0.284730	-1.691432	-0.885533	-0.067263	1.541149	0.514063	0.701227	-1.221392
5	-1.784229	0.547903	-0.418788	0.238425	-3.512096	1.495589	3.499186	1.201411	-0.648866	0.094526	0.321162	0.865918
...	...	...	...	...	...	...	...	...	...	...	...	...
2993	-1.784229	-0.269810	-1.470351	-0.121329	0.284730	0.066924	-0.184076	-0.826248	1.541149	0.514063	0.036114	-0.566550
2994	-1.784229	0.632080	1.114740	0.238425	0.284730	1.385692	0.455849	0.893923	-0.648866	0.514063	-1.484145	-0.975826
2995	0.560466	1.373633	-0.856939	-0.790472	0.284730	1.385692	0.208493	-0.976119	-0.648866	-0.464857	-1.104080	-0.239129
2996	0.560466	1.097054	-0.141293	-0.826448	0.284730	-1.032048	0.407855	-0.749638	1.541149	-0.744549	-0.343951	-0.075418
2997	0.560466	-0.069390	1.114740	0.103517	0.284730	1.385692	-0.270220	-0.858064	-0.648866	0.514063	-1.246604	-1.876235

2987 rows × 18 columns

In [33]:
```
x_train=data.iloc[:,0:11]
y_train=data.iloc[:,12]
x_test=data.iloc[:,0:11]
y_test=data.iloc[:,12]
```

In [34]:
```
x_train.head()
```

Out[34]:

	Gender	percentage	board	percentage.1	CollegeTier	Specialization	collegeGPA	CollegeCityID	CollegeCityTier	CollegeState	English
0	-1.784229	1.012878	-0.813124	0.867995	-3.512096	1.495589	0.286023	0.357861	1.541149	-1.723469	1.413848
1	0.560466	-2.073588	-0.813124	-0.885807	0.284730	-0.812254	-0.799389	0.295903	-0.648866	0.933600	-0.581491
2	0.560466	-0.036320	0.369883	0.973223	0.284730	-0.042973	-1.175961	0.810824	-0.648866	-0.325011	-0.153919
4	-1.784229	0.431660	-0.813124	0.058548	0.284730	-1.691432	-0.885533	-0.067263	1.541149	0.514063	0.701227
5	-1.784229	0.547903	-0.418788	0.238425	-3.512096	1.495589	3.499186	1.201411	-0.648866	0.094526	0.321162

In [35]:
```
y_test.head()
```

Out[35]:
```
0    0.402413
1   -0.356930
2    0.683124
4   -0.494621
5   -0.284169
Name: Domain, dtype: float64
```

In [36]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x_train, y_train, test_size = 0.2, random_state = 0)
```

In [37]:
```
print (x_train.shape)
print (x_test.shape)
```

```
(2389, 11)
(598, 11)
```

In [38]:
```
from sklearn.linear_model import LinearRegression
mlr = LinearRegression()
model_mlr = mlr.fit(x_train,y_train)
```

In [39]:
```
y_pred1 = model_mlr.predict(x_test)
```

In [40]:
```
MSE1 = mean_squared_error(y_test,y_pred1)
print('MSE is ', MSE1)
```

```
MSE is  0.9913519299159298
```

In [41]:
```
rf = RandomForestRegressor()
modelrf=rf.fit(x_train,y_train)
```

In [42]:
```
y_pred2 = modelrf.predict(x_test)
```

In [43]:
```
MSE2 = mean_squared_error(y_test,y_pred2)
print('MSE is ', MSE2)
```

```
MSE is  0.9359915497294157
```

In [45]:
```
y_pred1 = model_mlr.predict(x_test)
```

In [46]:
```
MSE3 = mean_squared_error(y_test,y_pred2)
print('LASSO is ', MSE3)
```

```
LASSO is  0.9359915497294157
```

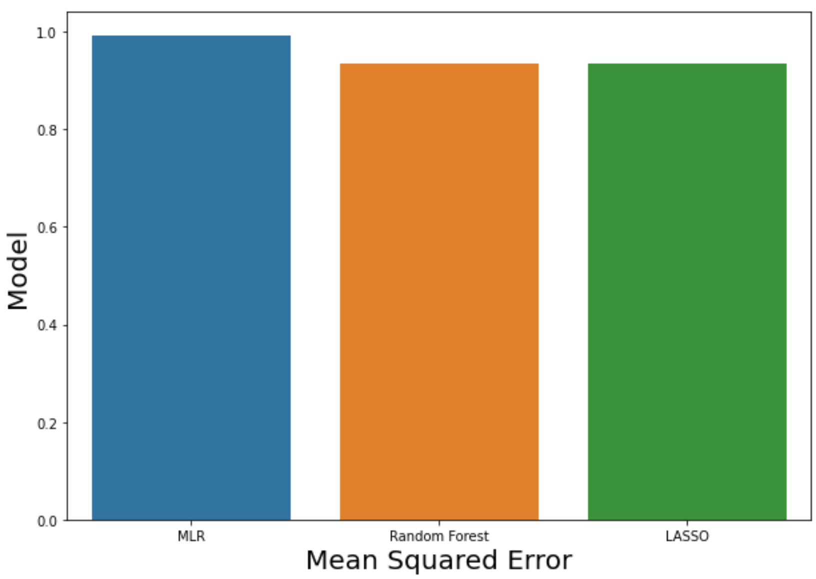
In [47]:
```
scores = [('MLR', MSE1),
 ('Random Forest', MSE2),('LASSO', MSE3)
 ]
```

In [48]:
```
MSE = pd.DataFrame(data = scores, columns=['Model', 'MSE Score'])
MSE
```

Out[48]:

	Model	MSE Score
0	MLR	0.991352
1	Random Forest	0.935992
2	LASSO	0.935992

In [49]:
```
MSE.sort_values(by=(['MSE Score']), ascending=False, inplace=True)
f, axe = plt.subplots(1,1, figsize=(10,7))
sns.barplot(x = MSE['Model'], y=MSE['MSE Score'], ax = axe)
axe.set_xlabel('Mean Squared Error', size=20)
axe.set_ylabel('Model', size=20)
plt.show()
```



In [ ]: