In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:

```
df = pd.read_csv("//home//yeshua//Documents//study//excel//Social_Network_Ads.csv")
```

In [3]:

```
df.head()
```

Out[3]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0

In [4]:

```
df.shape
```

Out[4]:

```
(400, 5)
```

In [5]:

```
df.describe(include="all")
```

Out[5]:

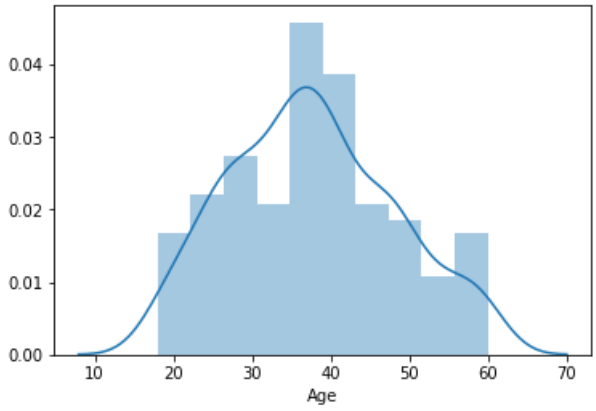
	User ID	Gender	Age	EstimatedSalary	Purchased
count	4.000000e+02	400	400.000000	400.000000	400.000000
unique	NaN	2	NaN	NaN	NaN
top	NaN	Female	NaN	NaN	NaN
freq	NaN	204	NaN	NaN	NaN
mean	1.569154e+07	NaN	37.655000	69742.500000	0.357500
std	7.165832e+04	NaN	10.482877	34096.960282	0.479864
min	1.556669e+07	NaN	18.000000	15000.000000	0.000000
25%	1.562676e+07	NaN	29.750000	43000.000000	0.000000
50%	1.569434e+07	NaN	37.000000	70000.000000	0.000000
75%	1.575036e+07	NaN	46.000000	88000.000000	1.000000
max	1.581524e+07	NaN	60.000000	150000.000000	1.000000

In [6]:

```
sns.distplot(df["Age"],bins=10,hist=True)
```

```
plt.show()
```

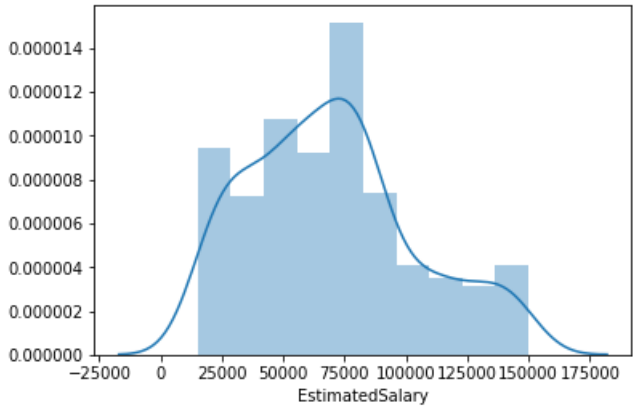
```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



In [7]:

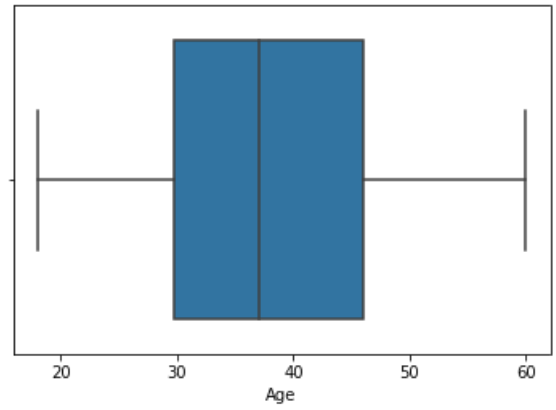
```
sns.distplot(df["EstimatedSalary"],bins=10,hist=True)
plt.show()
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



In [8]:

```
sns.boxplot(x=df["Age"])
plt.show()
```



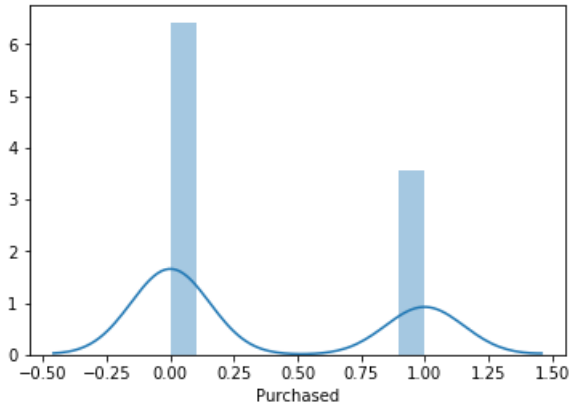
In [9]:

```
dfnum=df[["Age","EstimatedSalary","Purchased"]]
```

In [10]:

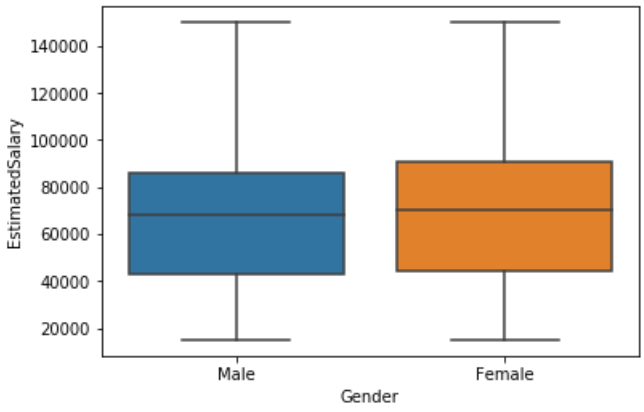
```
sns.distplot(dfnum["Purchased"],bins=10,hist=True)
plt.show()
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/matplotlib/axes/_axes.py:6462: UserWarning: The
'normed' kwarg is deprecated, and has been replaced by the 'density' kwarg.
  warnings.warn("The 'normed' kwarg is deprecated, and has been "
```



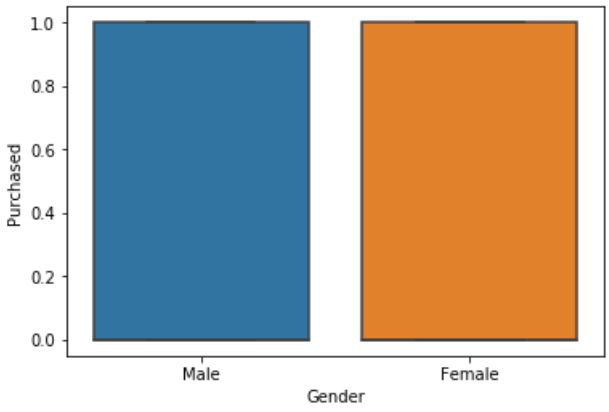
In [11]:

```
sns.boxplot(y=dfnum["EstimatedSalary"],x=df["Gender"])
plt.show()
```



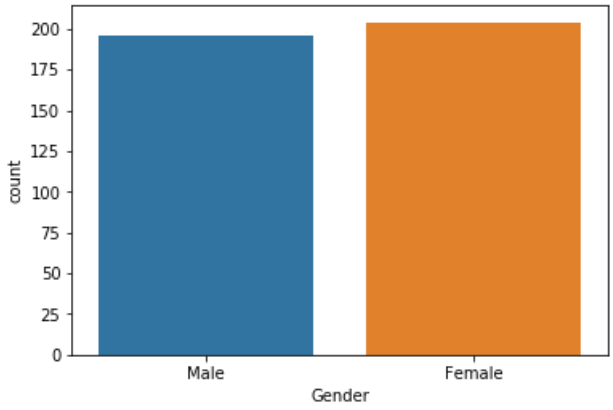
In [12]:

```
sns.boxplot(y=dfnum["Purchased"],x=df["Gender"])
plt.show()
```



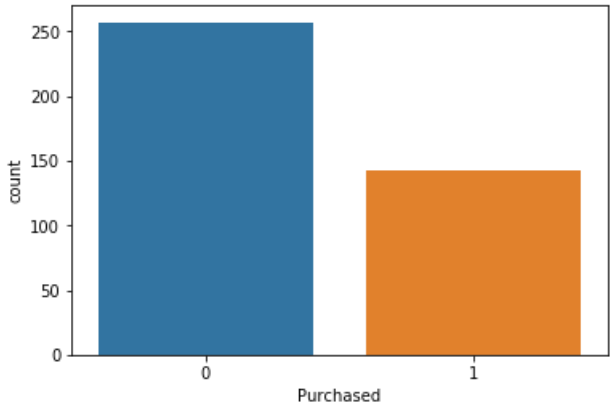
In [13]:

```
sns.countplot(x=df["Gender"])
plt.show()
```



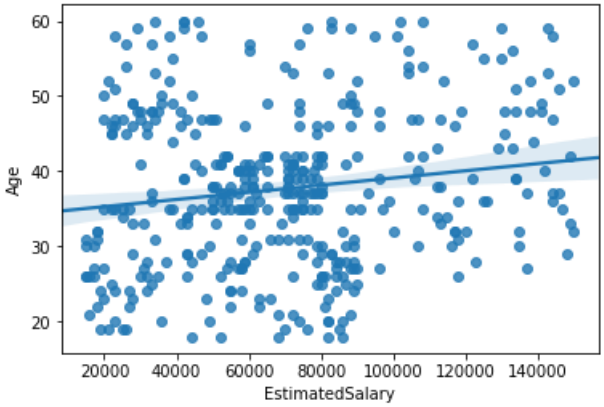
In [14]:

```
sns.countplot(x=df["Purchased"])
plt.show()
```



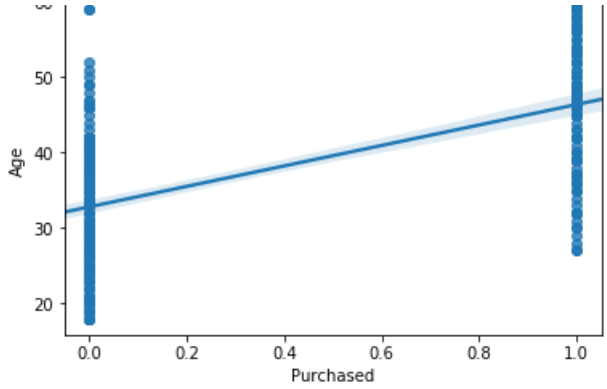
In [15]:

```
sns.regplot(x=df["EstimatedSalary"],y=df["Age"]),
plt.show()
```



In [16]:

```
sns.regplot(x=df["Purchased"],y=df["Age"]),
plt.show()
```



In [17]:

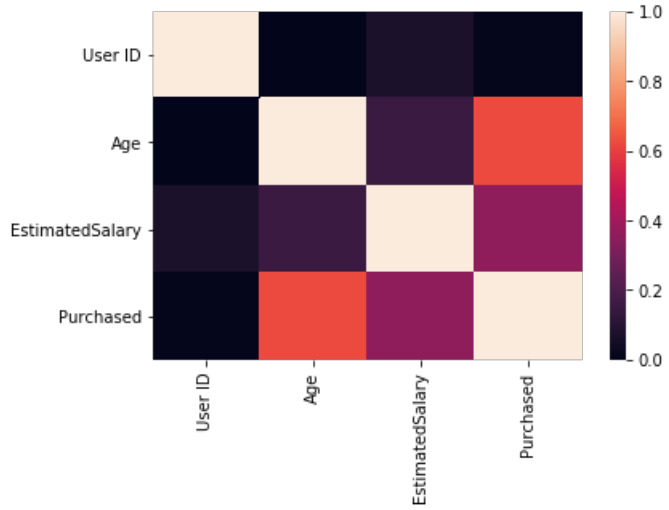
```
cor=df.corr()
cor
```

Out[17]:

	User ID	Age	EstimatedSalary	Purchased
User ID	1.000000	-0.000721	0.071097	0.007120
Age	-0.000721	1.000000	0.155238	0.622454
EstimatedSalary	0.071097	0.155238	1.000000	0.362083
Purchased	0.007120	0.622454	0.362083	1.000000

In [18]:

```
sns.heatmap(cor)
plt.show()
```



In [19]:

```
X = df.iloc[:, [2, 3]].values
y = df.iloc[:, 4].values

# Splitting the dataset into the Training set and Test set
from sklearn.cross_validation import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

```
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/cross_validation.py:41:
DeprecationWarning: This module was deprecated in version 0.18 in favor of the model_selection
module into which all the refactored classes and functions are moved. Also note that the interface
of the new CV iterators are different from that of this module. This module will be removed in 0.2
0.
  "This module will be removed in 0.20.", DeprecationWarning)
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475:
DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475:
DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
/home/yeshua/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:475:
DataConversionWarning: Data with input dtype int64 was converted to float64 by StandardScaler.
  warnings.warn(msg, DataConversionWarning)
```

In [20]:

```
# Fitting SVM to the Training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)
```

Out[20]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='linear',
  max_iter=-1, probability=False, random_state=0, shrinking=True,
  tol=0.001, verbose=False)
```

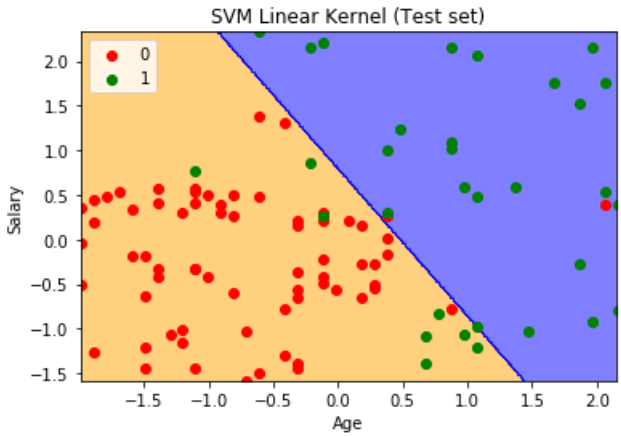
In [21]:

```
# Visualising the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test

aranged_ages = np.arange(start = X_set[:, 0].min(), stop = X_set[:, 0].max(), step = 0.01)
aranged_salaries = np.arange(start = X_set[:, 1].min(), stop = X_set[:, 1].max(), step = 0.01)

X1, X2 = np.meshgrid(aranged_ages, aranged_salaries)
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
     alpha = 0.5, cmap = ListedColormap(('orange', 'blue')))

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM Linear Kernel (Test set)')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.legend()
plt.show()
```



In [22]:

```
y_pred = classifier.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[66  2]
 [ 8 24]]
             precision    recall  f1-score   support

          0       0.89      0.97      0.93        68
          1       0.92      0.75      0.83        32

avg / total       0.90      0.90      0.90       100
```

In [23]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_set, y_pred, normalize=False)
```

Out[23]:

```
90
```

In [24]:

```
classifier = SVC(kernel = 'rbf', random_state = 0)
classifier.fit(X_train, y_train)
```

Out[24]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=0, shrinking=True,
  tol=0.001, verbose=False)
```

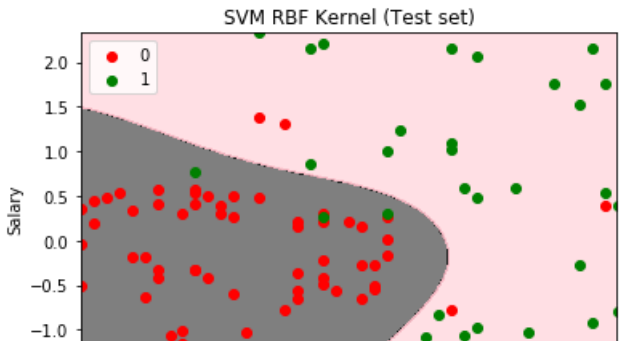
In [25]:

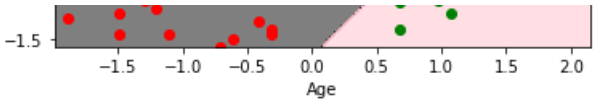
```
# Visualising the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test

aranged_ages = np.arange(start = X_set[:, 0].min(), stop = X_set[:, 0].max(), step = 0.01)
aranged_salaries = np.arange(start = X_set[:, 1].min(), stop = X_set[:, 1].max(), step = 0.01)

X1, X2 = np.meshgrid(aranged_ages, aranged_salaries)
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
     alpha = 0.5, cmap = ListedColormap(('black', 'pink')))

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM RBF Kernel (Test set)')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.legend()
plt.show()
```





In [26]:

```
y_pred = classifier.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[64  4]
 [ 3 29]]
             precision    recall  f1-score   support

          0       0.96      0.94      0.95        68
          1       0.88      0.91      0.89        32

avg / total       0.93      0.93      0.93       100
```

In [27]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_set, y_pred, normalize=False)
```

Out[27]:

```
93
```

In [28]:

```
classifier = SVC(kernel = 'poly', random_state = 0, degree = 3)
classifier.fit(X_train, y_train)
```

Out[28]:

```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovr', degree=3, gamma='auto', kernel='poly',
  max_iter=-1, probability=False, random_state=0, shrinking=True,
  tol=0.001, verbose=False)
```

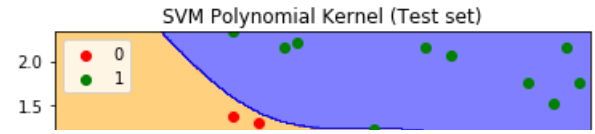
In [29]:

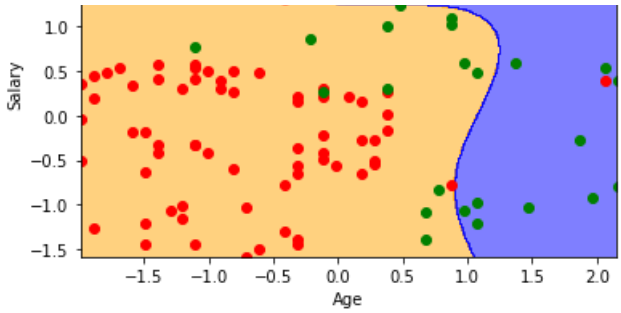
```
# Visualising the Test set results
from matplotlib.colors import ListedColormap
X_set, y_set = X_test, y_test

aranged_ages = np.arange(start = X_set[:, 0].min(), stop = X_set[:, 0].max(), step = 0.01)
aranged_salaries = np.arange(start = X_set[:, 1].min(), stop = X_set[:, 1].max(), step = 0.01)

X1, X2 = np.meshgrid(aranged_ages, aranged_salaries)
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
     alpha = 0.5, cmap = ListedColormap(('orange', 'blue')))

plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())
for i, j in enumerate(np.unique(y_set)):
    plt.scatter(X_set[y_set == j, 0], X_set[y_set == j, 1],
                c = ListedColormap(('red', 'green'))(i), label = j)
plt.title('SVM Polynomial Kernel (Test set)')
plt.xlabel('Age')
plt.ylabel('Salary')
plt.legend()
plt.show()
```





In [30]:

```
y_pred = classifier.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
[[67  1]
 [13 19]]
             precision    recall  f1-score   support

          0       0.84      0.99      0.91        68
          1       0.95      0.59      0.73        32

avg / total       0.87      0.86      0.85       100
```

In [31]:

```
from sklearn.metrics import accuracy_score
accuracy_score(y_set, y_pred, normalize=False)
```

Out[31]:

```
86
```