In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
data = pd.read_csv('Social_Network_Ads.csv')
```

In [3]:
```
data.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]:
```
data.tail()
```

Out[4]:

	User ID	Gender	Age	EstimatedSalary	Purchased
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

In [5]:
```
data.isnull()
```

Out[5]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
3	False	False	False	False	False
4	False	False	False	False	False
...	...	...	...	...	...
395	False	False	False	False	False
396	False	False	False	False	False
397	False	False	False	False	False
398	False	False	False	False	False
399	False	False	False	False	False

400 rows × 5 columns

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

Out[6]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

In [7]:
```
X = data.iloc[:,[2,3]]
```

In [8]:
```
y = data.iloc[:,-1]
```

In [9]:
```
from sklearn.model_selection import train_test_split
```

In [10]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, r
```

In [11]:
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
```

In [12]:
```
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Multivariate Linear Regression

In [14]:
```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train,y_train)
```

Out[14]:
```
▾ LinearRegression
LinearRegression()
```

In [15]:
```
pred = model.predict(X_test)
```

KNN

In [16]:
```
from sklearn.neighbors import KNeighborsClassifier
model1 = KNeighborsClassifier()
model1.fit(X_train,y_train)
```

Out[16]:
```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

In [17]:
```
pred1 = model1.predict(X_test)
```

Naive Bayes

In [18]:

```
from sklearn.naive_bayes import GaussianNB
model2 = GaussianNB()
model2.fit(X_train,y_train)
```

Out[18]:

```
▾ GaussianNB
GaussianNB()
```

In [19]:

```
pred2 = model2.predict(X_test)
```

Decision Tree

In [21]:

```
from sklearn.tree import DecisionTreeClassifier
model3 = DecisionTreeClassifier()
model3.fit(X_train,y_train)
```

Out[21]:

```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [22]:

```
pred3 = model3.predict(X_test)
```

SVM

In [23]:

```
from sklearn.svm import SVC
model4 = SVC()
model4.fit(X_train,y_train)
```

Out[23]:

```
▾ SVC
SVC()
```

In [24]:

```
pred4 = model4.predict(X_test)
```

Random Forest

In [25]:

```
from sklearn.ensemble import RandomForestClassifier
model5 = RandomForestClassifier()
model5.fit(X_train,y_train)
```

Out[25]:

```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [26]:

```
pred5 = model5.predict(X_test)
```

Log Rig

In [27]:

```
from sklearn.linear_model import LogisticRegression
model6 = LogisticRegression()
model6.fit(X_train,y_train)
```

Out[27]:

```
▾ LogisticRegression

LogisticRegression()
```

In [28]:

```
pred6 = model6.predict(X_test)
```

In [ ]:

In [ ]:

In [ ]:

```
from matplotlib.colors import ListedColormap
import numpy as np
import matplotlib.pyplot as plt
X_Set, Y_Set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_Set[:, 0].min() - 1, stop = X_Set[
                     np.arange(start = X_Set[:, 1].min() - 1, stop = X_Set[
Z = classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.sha
cmap = ListedColormap(['red', 'green'])
plt.contourf(X1, X2, Z, alpha=0.75, cmap=cmap)
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=cmap(i), labe
plt.title('Decision Tree (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

In [ ]: