A

MINI PROJECT REPORT ON

Support Vector Machine Algorithm on Social Media Network

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY,

PUNE. FOR

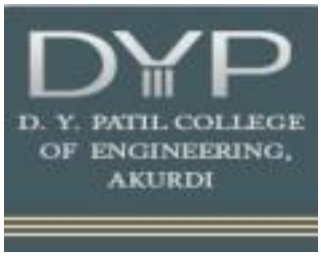
LAB PRACTICE III

Machine Learning

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

Name: Shweta Thikekar Exam Seat No: B150084340
Name: Devika Nehete Exam Seat No: B150084289
Name: Priyanka Kalokhe Exam Seat No: B150084261



DEPARTMENT OF COMPUTER ENGINEERING
D. Y. PATIL COLLEGE OF ENGINEERING AKURDI,

PUNE-44. SAVITRIBAI PHULE PUNE UNIVERSITY, 2021-22

SEM-II

INDEX

ABSTRACT

A social network dataset is a dataset containing the structural information of a social network. In the general case, a social network dataset consists of persons connected by edges. Social network datasets can represent friendship relationships or may be extracted from a social networking Web site. In this implementation of the SVM Classification model, we shall use a Social Network Advertisement dataset which consists of three columns. The first two columns are the independent variables, namely the 'Age ' and the 'Estimated Salary' and the last column is the dependent variable 'Purchased', which is in the binary format denoting whether the individual has bought the product (1) or not (0). In this problem, we have to build a SVM Classification model for a company that will classify whether a user of a particular age and with a particular salary will buy their given product or not.

INTRODUCTION

In simpler words we tell whether a user on Social Networking site after clicking the ad's displayed on the website, end's up buying the product or not. This could be really helpful for the company selling the product. Let's say that it's a car company which has paid the social networking site (For simplicity we'll assume its Facebook from now on) to display ads of its newly launched car. Now since the company relies heavily on the success of its newly launched car it would leave no stone unturned while trying to advertise the car. Well then what's better than advertising it on the most popular platform right now. But what if we only advertise it to the correct crowd. This could help in boosting sales as we will be showing the ad of the car only to selected crowd. So, this is where you come in.

The Car company has hired you as a Data Scientist to find out the correct crowd, to which you need to advertise the car and find out the people who are most likely to buy the car based on certain features which describe the type of users who have bought the car previously by clicking on the ad.

Problem Statement -

We have classifieds the data which will buy product or we have to make hike in our sale so that why we have applied the Support Vector Machine Algorithm for classification on a dataset In this problem, we have to build a SVM Classification model for a company that will classify whether a user of a particular age and with a particular salary will buy their given product or not.

Objectives and Scope-

- Provide hike in general product sale
- Effectively classifies the user based on particular age and with a particular salary
- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

SOFTWARE/HARDWARE REQUIREMENTS

Software Requirement Specifications-

1. Python
2. Visual Studio Code

Hardware Requirement Specification-

- **Operating System:**

  Product supports all OS and their versions

  For example:

  Windows OS, Mac OS, Linux mint, Ubuntu etc.

- **Android Devices:**

  Product is responsive and supports all android devices

- **Browser:**

  Product runs and supports all browser and their versions

  For example:

  Google chrome, Firefox etc.

Memory Requirement-

Memory, 2 GB minimum,

4 GB recommended Processor:

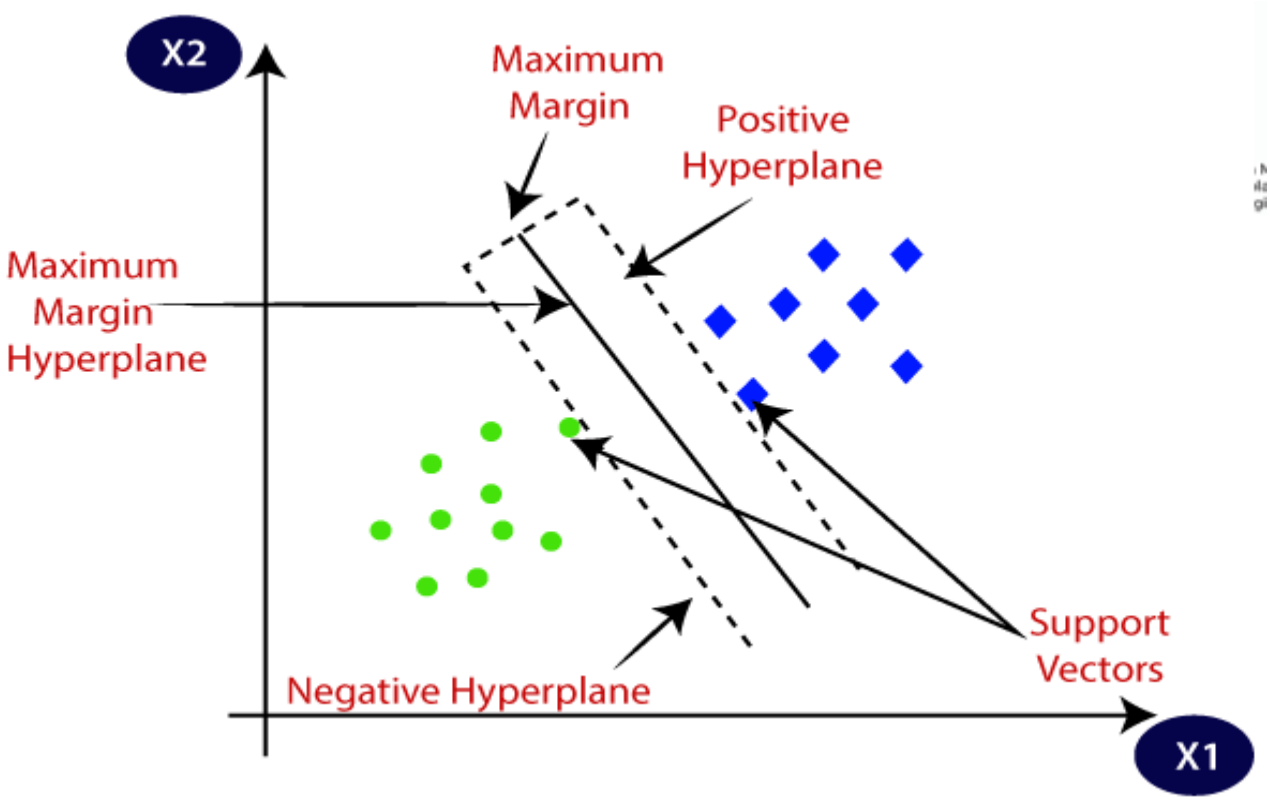
No GPU required and minimal processor works

ALGORITHM

Support vector machine algorithm, is linear. What makes the SVM algorithm stand out compared to other algorithms is that it can deal with classification problems using an SVM classifier and regression problems using an SVM regressor. However, one must remember that the SVM classifier is the backbone of the support vector machine concept and, in general, is the aptest algorithm to solve classification problems. Being a linear algorithm at its core can be imagined almost like a Linear or Logistic Regression. For example, an SVM classifier creates a line (plane or hyper-plane, depending upon the dimensionality of the data) in an N-dimensional space to classify data points that belong to two separate classes. It is also noteworthy that the original SVM classifier had this objective and was originally designed to solve binary classification problems, however unlike, say, linear regression that uses the concept of line of best fit, which is the predictive line that gives the minimum Sum of Squared Error (if using OLS Regression), or Logistic Regression that uses Maximum Likelihood Estimation to find the best fitting sigmoid curve, Support Vector Machines uses the concept of Margins to come up with predictions.

How Does Support Vector Machine Algorithm Work?

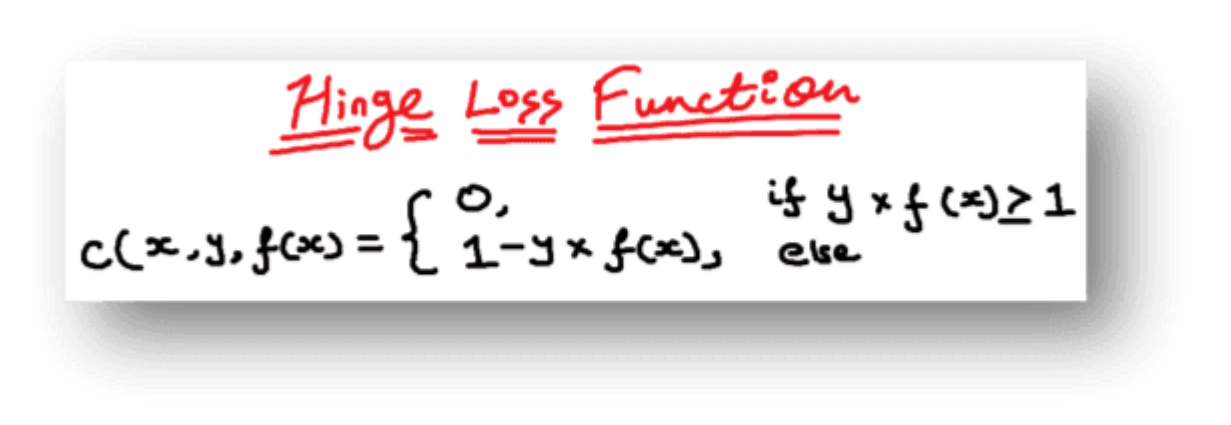
The best way to understand the SVM algorithm is by focusing on its primary type, the SVM classifier. The idea behind the SVM classifier is to come up with a hyper-lane in an N-dimensional space that divides the data points belonging to different classes. However, this hyper-pane is chosen based on margin as the hyperplane providing the maximum margin between the two classes is considered. These margins are calculated using data points known as Support Vectors. Support Vectors are those data points that are near to the hyper-plane and help in orienting it.



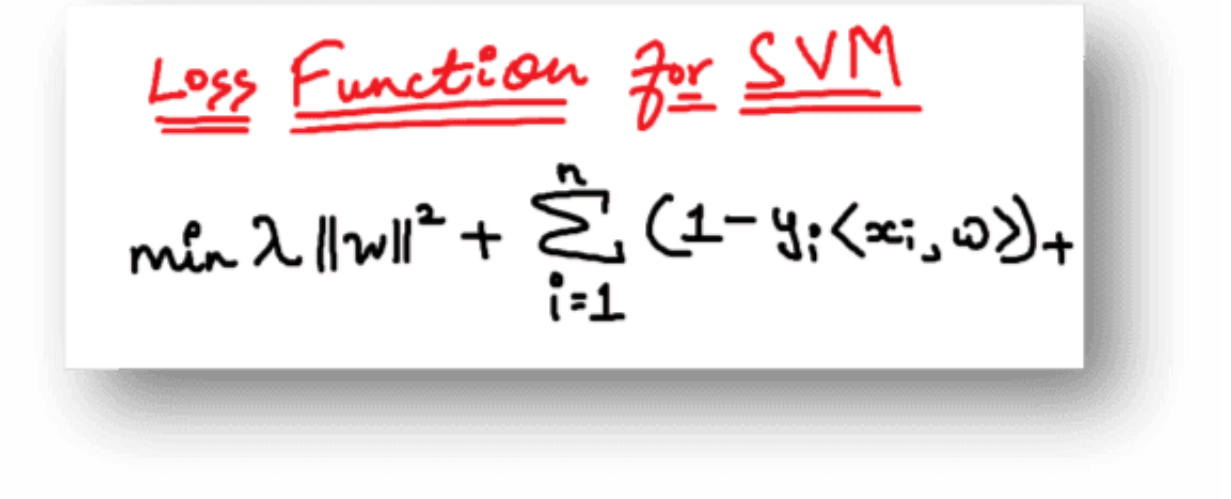
If the functioning of SVM classifier is to be understood mathematically then it can be understood in the following ways-

Step 1: SVM algorithm predicts the classes. One of the classes is identified as 1 while the other is identified as -1.

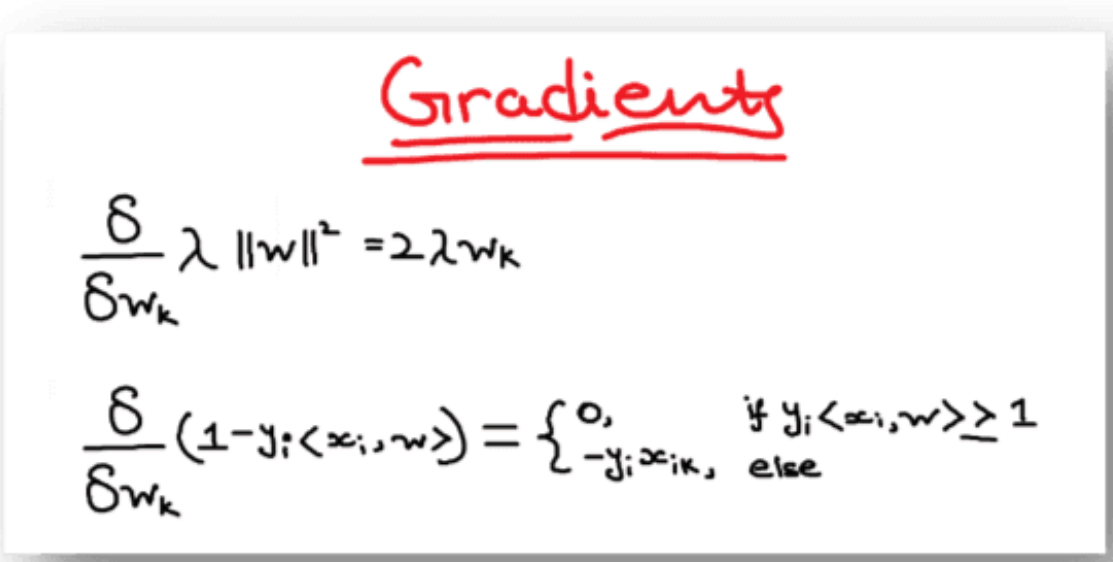
Step 2: As all machine learning algorithms convert the business problem into a mathematical equation involving unknowns. These unknowns are then found by converting the problem into an optimization problem.



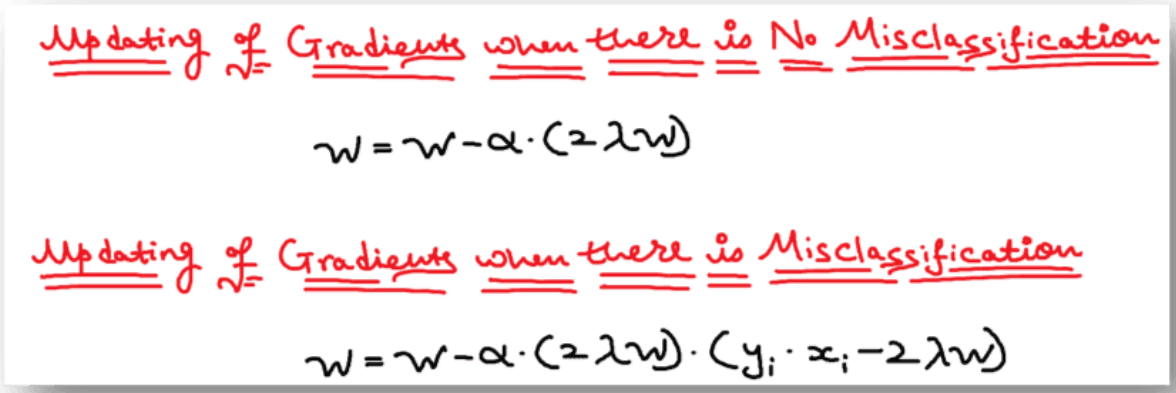
Step 3: For ease of understanding, this loss function can also be called a cost function whose cost is 0 when no class is incorrectly predicted. However, if this is not the case, then error/loss is calculated. The problem with the current scenario is that there is a trade-off between maximizing margin and the loss generated if the margin is maximized to a very large extent. To bring these concepts in theory, a regularization parameter is added.



Step 4: As is the case with most optimization problems, weights are optimized by calculating the gradients using advanced mathematical concepts of calculus viz. partial derivatives.



Step 5: The gradients are updated only by using the regularization parameter when there is no error in the classification while the loss function is also used when misclassification happens.



Step 6: The gradients are updated only by using the regularization parameter when there is no error in the classification, while the loss function is also used when misclassification happens.

Support Vector Machines – Implementation in Python

In Python, an SVM classifier can be developed using the sklearn library. The SVM algorithm steps include the following:

Step 1: Load the important libraries

- import pandas as pd
- import numpy as np
- import sklearn
- from sklearn import svm
- from sklearn.model_selection import train_test_split
- from sklearn import metrics

Step 2: Import dataset and extract the X variables and Y separately.

- df = pd.read_csv("*mydataset.csv"*)
- X = df.loc[:,['*Var_X1*','*Var_X2*','*Var_X3*','*Var_X4*']]
- Y = df[['*Var_Y*']]
-

Step 3: Divide the dataset into train and test

- X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size = 0.3, random_state=123)

Step 4: Initializing the SVM classifier model

- svm_clf = svm.SVC(kernel = 'linear')

Step 5: Fitting the SVM classifier model

- svm_clf.fit(X_train, y_train)

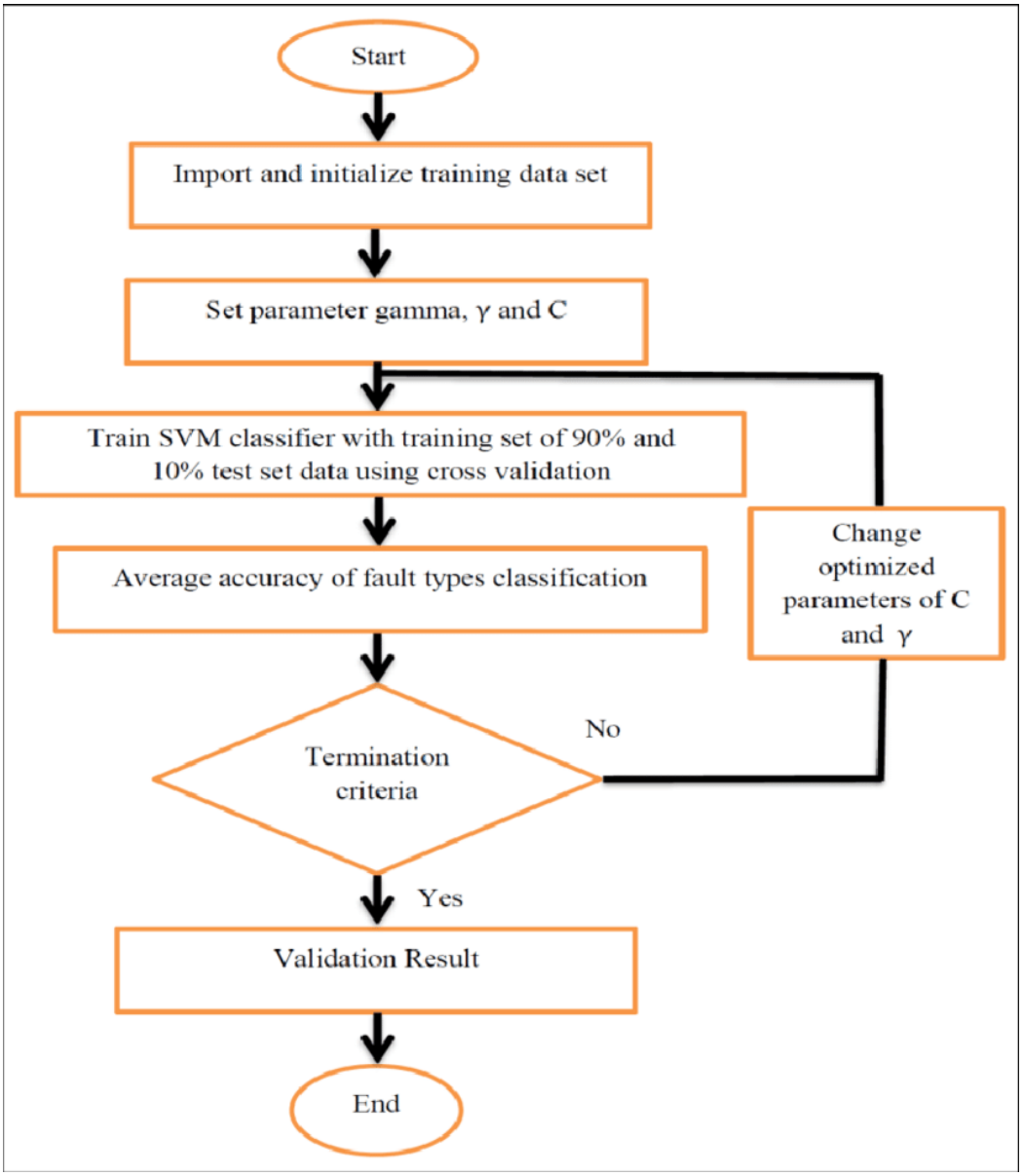
Step 6: Coming up with predictions

- y_pred_test = svm_clf.predict(X_test)

Step 7: Evaluating model's performance

- metrics.accuracy(y_test, y_pred_test)
- metrics.precision(y_test, y_pred_test)
- metrics.recall(y_test, y_pred_test)

Work Flow of Support Vector Machine Algorithm:

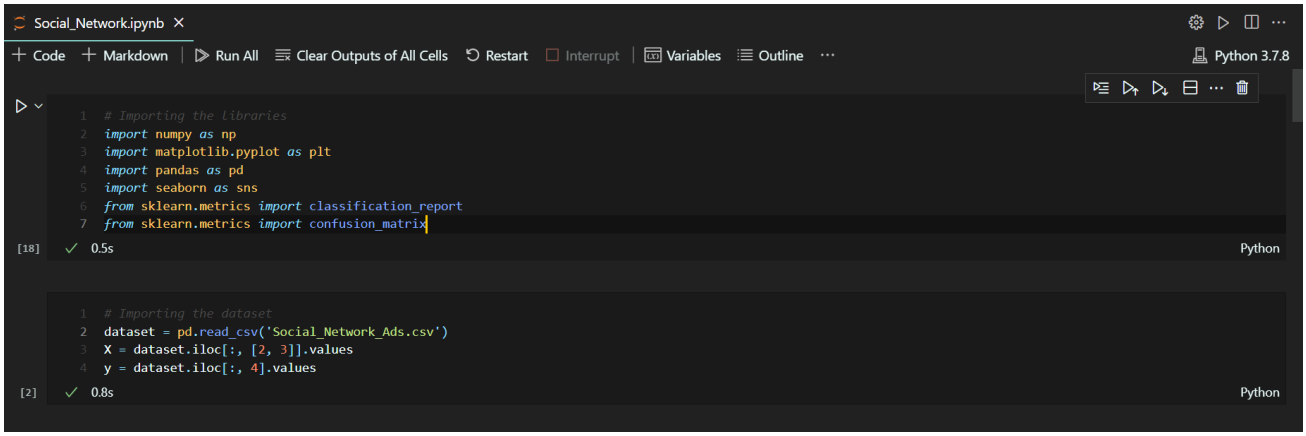

Start
Import and initialize training data set
Set parameter gamma, γ and C
Train SVM classifier with training set of 90% and 10% test set data using cross validation
Average accuracy of fault types classification
Termination criteria
No
Change optimized parameters of C and γ
Yes
Validation Result
End

Dataset Description-

In this implementation of the SVM Classification model, we shall use a Social Network Advertisement dataset which consists of three columns. The first two columns are the independent variables, namely the '***Age '***and the '***EstimatedSalary'*** and the last column is the dependent variable '***Purchased'***, which is in the binary format denoting whether the individual has bought the product (1) or not (0). In this problem, we have to build a SVM Classification model for a company that will classify whether a user of a particular age and with a particular salary will buy their given product or not.

RESULT

Working Module Screenshot-



Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
[3]  1 dataset.head()
     ✓ 0.6s                                                    Python
```

	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

```
[4]  1 dataset.tail(5)
     ✓ 0.3s                                                    Python
```

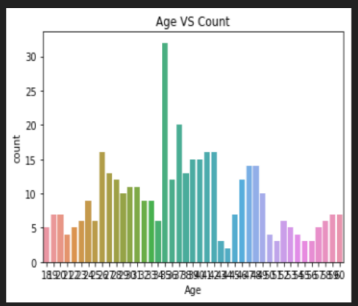
	User ID	Gender	Age	EstimatedSalary	Purchased
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
[38] 1 sns.countplot(x='Age',data=dataset)
     2
     3 plt.title('Age VS Count')
     4
     ✓ 0.4s                                                    Python
```

```
Text(0.5, 1.0, 'Age VS Count')
```



Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
x = dataset.iloc[:,2:32]
x.corr()
```

[39] ✓ 0.4s Python

	Age	EstimatedSalary	Purchased
Age	1.000000	0.155238	0.622454
EstimatedSalary	0.155238	1.000000	0.362083
Purchased	0.622454	0.362083	1.000000

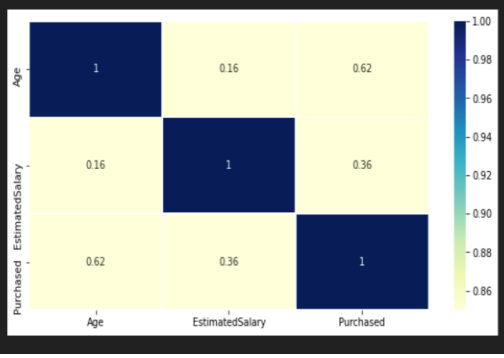
Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
plt.figure(figsize=(10,5))
sns.heatmap(x.corr(),vmin=0.85,vmax=1,annot=True,cmap='YlGnBu',linewidth=.5)
```

[45] ✓ 0.2s Python

```
<AxesSubplot:>
```



Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

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

[10] ✓ 1.5s Python

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

[11] ✓ 0.5s Python

```
# Fitting classifier to the Training set
from sklearn.svm import SVC
classifier = SVC(random_state=0) # for non-linear model use this parametre kernel='rbf'
classifier.fit(X_train, y_train)
```

[12] ✓ 0.3s Python

```
SVC(random_state=0)
```

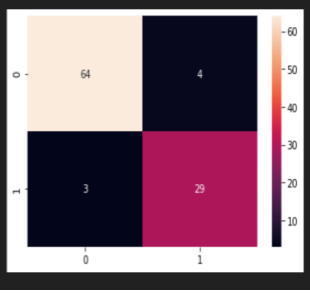
Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
# Making the Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, annot=True)
print(cm)
```

[30] ✓ 0.1s Python

```
[[64  4]
 [ 3 29]]
```



Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
# Making the Classification report
from sklearn.metrics import classification_report
cr = classification_report(y_test, y_pred)
print(cr)
```

[20] ✓ 0.3s Python

```
              precision    recall  f1-score   support

           0       0.96      0.94      0.95        68
           1       0.88      0.91      0.89        32

    accuracy                           0.93       100
   macro avg       0.92      0.92      0.92       100
weighted avg       0.93      0.93      0.93       100
```

Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8

```
# Visualising the Training set results
from matplotlib.colors import ListedColormap

X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(np.arange(start = X_set[:, 0].min() - 1, stop = X_set[:, 0].max() + 1, step = 0.01),
                     np.arange(start = X_set[:, 1].min() - 1, stop = X_set[:, 1].max() + 1, step = 0.01))
plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
             alpha = 0.75, cmap = ListedColormap(('red', 'green')))
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('Classifier (Training set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

[15] ✓ 3.7s Python

Social_Network.ipynb

Code | Markdown | Run All | Clear Outputs of All Cells | Restart | Interrupt | Variables | Outline | Python 3.7.8


```
#accuracy of SVM on Social Network dataset
from sklearn import svm
clf = svm.SVC(kernel='linear').fit(X_train,y_train)
from sklearn.metrics import accuracy_score
cp = clf.predict(X_test)
print(accuracy_score(y_test,cp)*100)
```

[29] ✓ 0.4s Python

```
90.0
```

Code:

https://github.com/shwetathikekar/SVM_Classification

CONCLUSION

we have successfully been able to build a SVM Classification Model that is able to predict if a person is going to buy a product based on his age and salary.

REFERENCES

1. https://www.analytixlabs.co.in/blog/introduction-support-vector-machine-algorithm/
2. https://medium.com/coinmonks/predicting-product-sales-through-ads-delivered-on-social-networking-sites-using-k-n-n-in-python-b9f955d184af
3. https://towardsdatascience.com/machine-learning-basics-support-vector-machine-svm-classification-205ecd28a09d