In [1]:

```
import numpy as np
import pandas as pd
import plotly
import plotly.figure_factory as ff
import plotly.graph_objs as go
from sklearn.linear_model import LogisticRegression, SGDClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)
```

In [2]:

```
data = pd.read_csv('task_b.csv')
data=data.iloc[:,1:]
```

In [3]:

```
data.head()
```

Out[3]:

	f1	f2	f3	y
0	-195.871045	-14843.084171	5.532140	1.0
1	-1217.183964	-4068.124621	4.416082	1.0
2	9.138451	4413.412028	0.425317	0.0
3	363.824242	15474.760647	1.094119	0.0
4	-768.812047	-7963.932192	1.870536	0.0

In [4]:

```
data.corr()['y']
```

Out[4]:

```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

In [5]:

```
data.std()
```

Out[5]:

```
f1      488.195035
f2    10403.417325
f3        2.926662
y         0.501255
dtype: float64
```

In [6]:

```
X=data[['f1','f2','f3']].values
Y=data['y'].values
print(X.shape)
print(Y.shape)
```

```
(200, 3)
(200,)
```

What if our features are with different variance

*** As part of this task you will observe how linear models work in case of data having feautres with different variance**
*** from the output of the above cells you can observe that var(F2)>>var(F1)>>Var(F3)**

> **Task1:**
    1. Apply Logistic regression(SGDClassifier with logloss) on 'data' and check the feature importance
    2. Apply SVM(SGDClassifier with hinge) on 'data' and check the feature importance

> **Task2:**
    1. Apply Logistic regression(SGDClassifier with logloss) on 'data' after standardization
       i.e standardization(data, column wise): (column-mean(column))/std(column) and check the feature importance
    2. Apply SVM(SGDClassifier with hinge) on 'data' after standardization
       i.e standardization(data, column wise): (column-mean(column))/std(column) and check the feature importance

In [7]:

```
def print_feature_importance(cfs):
  print("Co-efficients :")
  coefs = cfs[0]
  for i in [1,2,3]:
    print('f%d : %f'%(i,coefs[i-1]))
```

Task 1

Logistic regression(SGDClassifier with logloss)

In [8]:

```
clf = SGDClassifier(loss='log',random_state=0)
clf.fit(X,Y)
print_feature_importance(clf.coef_)
```

```
Co-efficients :
f1 : -1481.825952
f2 : 14346.683837
f3 : 10505.385694
```

SVM(SGDClassifier with hinge)

In [9]:

```
clf1 = SGDClassifier(loss='hinge',random_state=0)
clf1.fit(X,Y)
print_feature_importance(clf1.coef_)
```

```
Co-efficients :
f1 : 10127.953229
f2 : 14938.464405
f3 : 10232.765491
```

Observations

- Our model depends on variance as it involves finding optimal hyperplane. If one feature has high variance when compared to other features, then it would cause some sort of bias towards that feature which would cause incorrect results.
- In this case, as we do not standardsize the data, the feature importance is very large and hence cannot be used to draw any concrete conclusions about the feature importance. However, as f2 has highest variance, its feature importance is also signiificantly higher than the other 2 features.

Task 2

Standardize Data

In [10]:

```
df = data.copy()

for col in df.columns[:-1]:
  df[col] = df[col].apply(lambda x : (x-np.mean(df[col]))/np.std(df[col]))

X_std=df[['f1','f2','f3']].values
```

In [11]:

```
# print(data.head(1))
# print(df.head(1))
```

Logistic regression(SGDClassifier with logloss)

In [12]:

```
clf = SGDClassifier(loss='log',random_state=0)
clf.fit(X_std,Y)
print_feature_importance(clf.coef_)
```

```
Co-efficients :
f1 : 1.679927
f2 : 0.452358
f3 : 9.618069
```

SVM(SGDClassifier with hinge)

In [13]:

```
clf = SGDClassifier(loss='hinge',random_state=0)
clf.fit(X_std,Y)
print_feature_importance(clf.coef_)
```

```
Co-efficients :
f1 : 0.087239
f2 : 0.465957
f3 : 9.980700
```

After standardization

- As our data is standardized we are sure that our data is centered and scaled properly and hence we can interpret our feature importances.
- We can see that feature f3 highest contribution towards class 1 followed by f1. Hence, f1 and f3 give more weightage for a data point to be classified as class = 1. Whereas f2 emphasis data point belong to class 0

In [13]:

```

```