In [58]:
```
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
from sklearn.linear_model import SGDClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler
from sklearn import svm
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, ipl
init_notebook_mode(connected=True)
```

In [3]:
```
data = pd.read_csv('task_b.csv')
data=data.iloc[:,1:]
```

In [4]:
```
data.head()
```

Out[4]:

	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 [5]:
```
data.corr()['y']
```

Out[5]:
```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

In [6]:
```
data.std()
```

Out[6]:
```
f1      488.195035
f2    10403.417325
f3        2.926662
y         0.501255
dtype: float64
```

In [32]:
```
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

Make sure you write the observations for each task, why a particular feautre got more importance than others

Task 1:

In [55]:
```
model = LogisticRegression()
model.fit(X, Y)
importance = abs(model.coef_[0])
for i,j in enumerate(importance):
    print("Feature:", i, "  Importance:", j)
```

```
Feature: 0   Importance: 0.0008896381130512665
Feature: 1   Importance: 1.041694609480307e-05
Feature: 2   Importance: 1.9566801565321608
```

In [60]:
```
clf = svm.SVC(kernel='linear')
clf.fit(X,Y)
importance = abs(clf.coef_[0])
for i,j in enumerate(importance):
    print("Feature:", i, "  Importance:", j)
```

```
Feature: 0   Importance: 0.003200025217765301
Feature: 1   Importance: 0.0007265928873039229
Feature: 2   Importance: 15.368257657386815
```

Task 2:

Standardizing and transforming data

In [33]:
```
scalar = StandardScaler()
scalar.fit(X, Y)
X_transform = scalar.transform(X)
print("After transformation")
print(X_transform.shape, Y.shape)
```

```
After transformation
(200, 3) (200,)
```

In [61]:
```
model = LogisticRegression()
model.fit(X_transform, Y)
importance = abs(model.coef_[0])
for i,j in enumerate(importance):
    print("Feature:", i, "  Importance:", j)
```

```
Feature: 0   Importance: 0.24105482944744724
Feature: 1   Importance: 0.07588844908289771
Feature: 2   Importance: 3.903568993476378
```

In [62]:
```
clf = svm.SVC(kernel = 'linear')
clf.fit(X_transform,Y)
importance = abs(clf.coef_[0])
for i,j in enumerate(importance):
    print("Feature:", i, "  Importance:", j)
```

```
Feature: 0   Importance: 0.21568590826123657
Feature: 1   Importance: 0.0746398125056571
Feature: 2   Importance: 2.932488560228915
```

Observation:

- In task 1, both Logistic Regression and SVM gives feature 3 as more important than others.
- In task 2, similarly, even after standardization, feature 3 has more importance than other features.