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 SGDClassifier
from sklearn.linear_model import LogisticRegression
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

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

Task 1

Logistic Regression

In [7]:
```
LR_clf=SGDClassifier(loss= 'log', random_state = 15)
```

In [8]:
```
LR_clf.fit(X,Y)
```

Out[8]:
```
SGDClassifier(loss='log', random_state=15)
```

In [9]:
```
LR_clf.coef_
```

Out[9]:
```
array([[  3925.14601273, -16033.05764291,  10502.94022174]])
```

In [10]:
```
feature_weights = {}
for feature, weights in zip(['f1','f2','f3'],LR_clf.coef_[0]):
    feature_weights[feature]=weights
```

In [11]:
```
print(feature_weights)
```

```
{'f1': 3925.1460127265923, 'f2': -16033.057642911479, 'f3': 10502.94022174132}
```

Observations:

1. The most important features for Logistic Regression are in the order f2>f3>f1
2. f2 is the most important feature as it has the highest variance among the three features.
3. f3 is the second important feature as it has the highest correlation among the three features

SVM

In [12]:
```
svm_clf = SGDClassifier(loss= 'hinge', random_state = 15)
svm_clf.fit(X,Y)
```

Out[12]:
```
SGDClassifier(random_state=15)
```

In [13]:
```
svm_clf.coef_
```

Out[13]:
```
array([[-1441.65036452, -3083.88512888, 10638.5348658 ]])
```

In [14]:
```
feature_weights = {}
for feature, weights in zip(['f1','f2','f3'],svm_clf.coef_[0]):
    feature_weights[feature]=weights
```

In [15]:
```
print(feature_weights)
```

```
{'f1': -1441.6503645194791, 'f2': -3083.8851288782294, 'f3': 10638.534865801403}
```

Observations:

1. The most important feature for SVM are in the order f3>f2>f1
2. f3 is the most important feature as it has the highest correlation among the three features
3. f2 is the second most important feature as it has the highest variance among the three features
4. SVM seems to be less impacted by variance of the features compared to Logistic Regression

Task 2:

Logistic Regression

In [16]:
```
standard_data=StandardScaler().fit_transform(data[['f1','f2','f3']])
```

In [17]:
```
std_LR_clf=SGDClassifier(loss= 'log', random_state = 15)
```

In [18]:
```
std_LR_clf.fit(standard_data,Y)
```

Out[18]:
```
SGDClassifier(loss='log', random_state=15)
```

In [19]:
```
std_LR_clf.coef_
```

Out[19]:
```
array([[-0.29741788, -0.66973479, 10.35436789]])
```

In [20]:
```
feature_weights = {}
for feature, weights in zip(['f1','f2','f3'],std_LR_clf.coef_[0]):
    feature_weights[feature]=weights
```

In [21]:
```
print(feature_weights)
```

```
{'f1': -0.2974178841831834, 'f2': -0.6697347941199512, 'f3': 10.354367890268982}
```

Observations:

1. The most importand features for Logistic Regression on Standardized data is of the order f3>f2>f1
2. After standardization, the variance of all the features have become equal. So, the most important feature has become f3 which has the highest correlation.

SVM

In [22]:
```
std_svm_clf=SGDClassifier(loss= 'hinge', random_state = 15)
```

In [23]:
```
std_svm_clf.fit(standard_data,Y)
```

Out[23]:
```
SGDClassifier(random_state=15)
```

In [24]:
```
std_svm_clf.coef_
```

Out[24]:
```
array([[ 2.23347737,  0.46842383, 22.39791493]])
```

In [25]:
```
feature_weights = {}
for feature, weights in zip(['f1','f2','f3'],std_svm_clf.coef_[0]):
    feature_weights[feature]=weights
```

In [26]:
```
print(feature_weights)
```

```
{'f1': 2.233477374028122, 'f2': 0.46842383180192404, 'f3': 22.397914928450106}
```

Observations:

1. The most importand features for SVM on Standardized data is of the order f3>f1>f2
2. The most important feature for SVM is f3 which has the highest correlation
3. The second most important feature for SVM is f1 which has second highest correlation
4. f2 which was the second important feature in non-standardized data because of its high variance has become least important feature after standardization