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

In [3]:
```python
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]:
```python
data.corr()['y']
```

Out[4]:
```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

In [5]:
```python
data.std()
```

Out[5]:
```
f1      488.195035
f2    10403.417325
f3        2.926662
y         0.501255
dtype: float64
```

In [6]:
```python
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

Build a Model Without Feature Standardization

In [125…

```python
# Logistic Regression with SGD Classifier
from sklearn.linear_model import SGDClassifier
logistic = SGDClassifier(loss ='log')
logistic.fit(X,Y)
#coeff
coeff = logistic.coef_
features = data.columns
for feature,coef in zip(features[:-1],coeff[0]):
    print("{0} -coefficient : {1}".format(feature,coef))
    print('{0} - absolute weight : {1}'.format(feature,abs(coef)) )
    print("="*50)
print('Score :{}'.format(logistic.score(X,Y)))
```

```
f1 -coefficient : 1953.2570698207414
f1 - absolute weight : 1953.2570698207414
==================================================
f2 -coefficient : -16467.82281633211
f2 - absolute weight : 16467.82281633211
==================================================
f3 -coefficient : 10594.926936428314
f3 - absolute weight : 10594.926936428314
==================================================
Score :0.525
```

In [126…

```python
# Logistic Regression with SGD Classifier
from sklearn.linear_model import SGDClassifier
svm = SGDClassifier(loss ='hinge')
svm.fit(X,Y)
#coeff
coeff = svm.coef_
features = data.columns
for feature,coef in zip(features[:-1],coeff[0]):
    print("{0} -coefficient : {1}".format(feature,coef))
    print('{0} - absolute weight : {1}'.format(feature,abs(coef)) )
    print("="*50)
print('Score :{}'.format(svm.score(X,Y)))
```

```
f1 -coefficient : 10260.587068136601
f1 - absolute weight : 10260.587068136601
==================================================
f2 -coefficient : -2091.97196298414
f2 - absolute weight : 2091.97196298414
==================================================
f3 -coefficient : 11098.507935917825
f3 - absolute weight : 11098.507935917825
==================================================
Score :0.545
```

Observation

Feature Important is useful to Decide the Class Label

Weight tells which Feature Important for predicting the class label In this Situation Feature f1 and f3 is weight Positive Weights,F3 Feature is Highest Positive Weight, so Feature F3 is More Important for deciding the class label as positive,f1 is less important comparing to f3

The feature f2 is Negative is used to decide the class label negative,because f2 weight is negative

AS We Observe the Training Performance for Models Logistic Regession and SVM, The Models Not Perform Good in Training time.as Considerd as Average Because of High Varience of Features has high Varience. if the Varience is High distance from the points is increased Lot of Miss classification happens.

In [129…

```
#Standardized_data
X_std = StandardScaler().fit_transform(X)
logistic = SGDClassifier(loss ='log')
logistic.fit(X_std,Y)
#coeff
coeff = logistic.coef_
features = data.columns
for feature,coef in zip(features[:-1],coeff[0]):
    print("{0} -coefficient : {1}".format(feature,coef))
    print('{0} - absolute weight : {1}'.format(feature,abs(coef)) )
    print("="*50)
print('Score :{}'.format(logistic.score(X_std,Y)))
```

```
f1 -coefficient : -1.803327383728873
f1 - absolute weight : 1.803327383728873
==================================================
f2 -coefficient : 3.286491308755802
f2 - absolute weight : 3.286491308755802
==================================================
f3 -coefficient : 11.127307064509534
f3 - absolute weight : 11.127307064509534
==================================================
Score :0.91
```

In [127…

```
# Standardizing Svm

svm = SGDClassifier(loss ='hinge')
svm.fit(X_std,Y)
#coeff
coeff = svm.coef_
features = data.columns
for feature,coef in zip(features[:-1],coeff[0]):
    print("{0} -coefficient : {1}".format(feature,coef))
    print('{0} - absolute weight : {1}'.format(feature,abs(coef)) )
    print("="*50)
print('Score :{}'.format(svm.score(X_std,Y)))
```

```
f1 -coefficient : -7.56952434037253
f1 - absolute weight : 7.56952434037253
==================================================
f2 -coefficient : -0.9831145132336695
f2 - absolute weight : 0.9831145132336695
==================================================
f3 -coefficient : 21.67907853796233
f3 - absolute weight : 21.67907853796233
==================================================
Score :0.905
```

Observation

After Standardizing the Features weights is completely changed,Because all the Fetures distribution in the range -1 to 1 When We apply the standardized features both the model Perform good we got Better Accuracy, So High Varience is Affect the model Performance if the varience is high model is too underfit

In Logitic Regression f2 and f3 is weight is large positive so impact to the model lot of large predicting poitive,feature f1 is impact to predict the class label as negative

SVM f1 and f2 is weight is large positive so impact to the model lot of large predicting poitive,feature f1 is impact to predict the class label as negative

Conclusion

1.Feature Scaling is Important Before Build the Model,it affect the Model Performance,we saw Logistic regression and Svm Not perform well in training well

2.Model Performance and Feature Importance Changed After Standardization

3.feature has More Weight is considered as important features