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

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

Out[42]:

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

Out[43]:
```
f1    0.067172
f2   -0.017944
f3    0.839060
y     1.000000
Name: y, dtype: float64
```

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

Out[44]:
```
f1      488.195035
f2    10403.417325
f3        2.926662
y         0.501255
dtype: float64
```

In [ ]:
```
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 [ ]:
```
clf_logistic = LogisticRegression(penalty='l2')
clf_logistic.fit(X,Y)
```

Out[46]:
```
LogisticRegression()
```

In [ ]:
```
mean_score= clf_logistic.score(X,Y)
```

In [ ]:
```
featureImportance= clf_logistic.coef_[0]
```

Type *Markdown* and LaTeX: α^2

In [ ]:
```
print("Below are the results which we have got")
print("Accuracy of Logistic Regression is ",mean_score)
for id in range(featureImportance.size):
  print(f"f{id} has coefficient value of {abs(featureImportance[id])}")
```

```
Below are the results which we have got
Accuracy of Logistic Regression is  0.93
f0 has coefficient value of 0.0008896381130511962
f1 has coefficient value of 1.0416946094804762e-05
f2 has coefficient value of 1.9566801565320402
```

In [ ]:
```
# SVM classfier
from sklearn.svm import LinearSVC
clf_LinearSVM = LinearSVC(penalty='l2', loss='hinge', max_iter=100000, tol=10e-5)
```

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

```
/usr/local/lib/python3.7/dist-packages/sklearn/svm/_base.py:1208: ConvergenceWa
rning:

Liblinear failed to converge, increase the number of iterations.
```

Out[51]:
```
LinearSVC(loss='hinge', max_iter=100000)
```

In [ ]:
```
mean_score = clf_LinearSVM.score(X,Y)
```

In [ ]:
```
featureImportance = clf_LinearSVM.coef_[0]
```

In [ ]:
```
print("Below are the results which we have got")
print("Accuracy of Logistic Regression is ",mean_score)
for id in range(featureImportance.size):
  print(f"f{id} has coefficient value of {abs(featureImportance[id])}")
```

```
Below are the results which we have got
Accuracy of Logistic Regression is  0.82
f0 has coefficient value of 0.00013121116182661445
f1 has coefficient value of 9.290952865687918e-06
f2 has coefficient value of 0.19403950783809396
```

Observation From Experiment 1

1. In addition to fitting well, logistic regression is also accurate.
2. More than 100000 iterations of SVM linear with hinge loss failed to converge.
3. Classifier behavior is affected by the high variance of data.
4. Feature importance is more evenly divided in linear regression comparison to svm.

In [ ]:
```
# Scale the feature to zero mean Unit variance
ss = StandardScaler()
X = ss.fit_transform(X)
```

In [ ]:
```
# Logistic Regression on Standaradized data
clf_logistic_ss = LogisticRegression(penalty='l2')
```

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

Out[57]:
```
LogisticRegression()
```

In [ ]:
```
# Feature importance
featureImportance_standard = clf_logistic_ss.coef_[0]
```

In [ ]:
```
mean_score_stanrard = clf_logistic_ss.score(X,Y)
```

In [ ]:
```
print("Below are the results which we have got")
print("Accuracy of Logistic Regression is ",mean_score_stanrard)
for id in range(featureImportance.size):
  print(f"f{id} has coefficient value of {abs(featureImportance[id])}")
```

```
Below are the results which we have got
Accuracy of Logistic Regression is  0.93
f0 has coefficient value of 0.00013121116182661445
f1 has coefficient value of 9.290952865687918e-06
f2 has coefficient value of 0.19403950783809396
```

In [ ]:
```
# Run Same Experiment on SVM
# SVM classfier
svmLinear_standard = LinearSVC(penalty='l2', loss='hinge', max_iter=300, tol=10e-
svmLinear_standard.fit(X,Y)
score_svm_standard = svmLinear_standard.score(X,Y)
featureImportance = svmLinear_standard.coef_[0]
```

In [ ]:
```
print("Below are the results which we have got")
print("Accuracy of Logistic Regression is ",score_svm_standard)
for id in range(featureImportance.size):
  print(f"f{id} has coefficient value of {abs(featureImportance[id])}")
```

```
Below are the results which we have got
Accuracy of Logistic Regression is  0.925
f0 has coefficient value of 0.21573733233050676
f1 has coefficient value of 0.0752051991613595
f2 has coefficient value of 2.933692852911257
```

Observation from Experiment 2

1. Logistic regression fits quiet well and accuracy is also good with standarized features.
2. In maximum 300 iterations, SVM converges.
3. SVM Linear convergence was sped up after standardization as well as accuracy raised to 0.925.

Observation from Both Experiment

1. As for accuracy, it is also very good with and without standardization of the features in logistic regression.
2. In SVM linear, variance of features in data seems to be highly sensitive. Therefore, standardizing the data helps overcome the high variance nature of the dataset and helps improve the classifier.

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