```
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)

from google.colab import files
uploaded = files.upload()
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

Saving task_b.csv to task_b.csv

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


data.head()
```

	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

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

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

```
data.std()
```

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

```
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 fe**
* **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 fea
2. Apply SVM(SGDClassifier with hinge) on 'data' and check the feature importance

> **Task2**:

1. Apply Logistic regression(SGDClassifier with logloss) on 'data' after standardiza
   i.e standardization(data, column wise): (column-mean(column))/std(column) and che
2. Apply SVM(SGDClassifier with hinge) on 'data' after standardization
   i.e standardization(data, column wise): (column-mean(column))/std(column) and che

```
from sklearn import linear_model
clf_lr_nstd = linear_model.SGDClassifier(eta0=0.0001, alpha=0.0001, loss='log', random_sta
clf_lr_nstd.fit(X,Y)
print(clf_lr_nstd.coef_)
print(clf_lr_nstd.intercept_)
```

```
-- Epoch 1
Norm: 1.08, NNZs: 3, Bias: -0.001751, T: 200, Avg. loss: 2516.147588
Total training time: 0.00 seconds.
-- Epoch 2
Norm: 0.61, NNZs: 3, Bias: -0.001551, T: 400, Avg. loss: 2621.694380
Total training time: 0.00 seconds.
-- Epoch 3
Norm: 0.35, NNZs: 3, Bias: -0.001850, T: 600, Avg. loss: 3285.222158
Total training time: 0.00 seconds.
-- Epoch 4
Norm: 0.64, NNZs: 3, Bias: -0.003527, T: 800, Avg. loss: 3142.216822
Total training time: 0.00 seconds.
-- Epoch 5
Norm: 0.48, NNZs: 3, Bias: -0.004027, T: 1000, Avg. loss: 3009.886714
Total training time: 0.00 seconds.
-- Epoch 6
Norm: 1.40, NNZs: 3, Bias: -0.003523, T: 1200, Avg. loss: 3032.001946
Total training time: 0.00 seconds.
Convergence after 6 epochs took 0.00 seconds
```

```
[[ 0.37170471 -1.34463853  0.12669033]]
[-0.00352309]
```

```
from sklearn import linear_model
clf_SVM_Nstd = linear_model.SGDClassifier(eta0=0.0001, alpha=0.0001, loss='hinge', random_
clf_SVM_Nstd.fit(X,Y)
print(clf_SVM_Nstd.coef_)
print(clf_SVM_Nstd.intercept_)
```

```
-- Epoch 1
Norm: 0.61, NNZs: 3, Bias: -0.001600, T: 200, Avg. loss: 2634.084615
Total training time: 0.00 seconds.
-- Epoch 2
Norm: 0.68, NNZs: 3, Bias: -0.001100, T: 400, Avg. loss: 2593.136418
Total training time: 0.00 seconds.
-- Epoch 3
Norm: 0.76, NNZs: 3, Bias: -0.000900, T: 600, Avg. loss: 3308.216351
Total training time: 0.00 seconds.
-- Epoch 4
Norm: 0.77, NNZs: 3, Bias: -0.002700, T: 800, Avg. loss: 3155.085896
Total training time: 0.00 seconds.
-- Epoch 5
Norm: 0.93, NNZs: 3, Bias: -0.002800, T: 1000, Avg. loss: 3080.501847
Total training time: 0.00 seconds.
-- Epoch 6
Norm: 0.43, NNZs: 3, Bias: -0.002700, T: 1200, Avg. loss: 3011.887174
Total training time: 0.00 seconds.
-- Epoch 7
Norm: 0.69, NNZs: 3, Bias: -0.002200, T: 1400, Avg. loss: 3002.132514
Total training time: 0.00 seconds.
Convergence after 7 epochs took 0.00 seconds
[[ 0.38249139 -0.55764501  0.15407861]]
[-0.0022]
```

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_std = scaler.fit_transform(X)
clf_LR_std = linear_model.SGDClassifier(eta0=0.0001, alpha=0.0001, loss='log', random_stat
clf_LR_std.fit(X_std,Y)
print(clf_LR_std.coef_)
print(clf_LR_std.intercept_)
```

```
-- Epoch 1
Norm: 0.01, NNZs: 3, Bias: 0.000001, T: 200, Avg. loss: 0.691431
Total training time: 0.00 seconds.
-- Epoch 2
Norm: 0.02, NNZs: 3, Bias: 0.000002, T: 400, Avg. loss: 0.687922
Total training time: 0.00 seconds.
-- Epoch 3
Norm: 0.03, NNZs: 3, Bias: 0.000002, T: 600, Avg. loss: 0.684449
Total training time: 0.00 seconds.
-- Epoch 4
Norm: 0.03, NNZs: 3, Bias: 0.000003, T: 800, Avg. loss: 0.681011
Total training time: 0.01 seconds.
-- Epoch 5
Norm: 0.04, NNZs: 3, Bias: 0.000003, T: 1000, Avg. loss: 0.677608
Total training time: 0.01 seconds.
-- Epoch 6
```

```
Norm: 0.05, NNZs: 3, Bias: 0.000003, T: 1200, Avg. loss: 0.674240
Total training time: 0.02 seconds.
-- Epoch 7
Norm: 0.06, NNZs: 3, Bias: 0.000003, T: 1400, Avg. loss: 0.670905
Total training time: 0.02 seconds.
-- Epoch 8
Norm: 0.07, NNZs: 3, Bias: 0.000003, T: 1600, Avg. loss: 0.667605
Total training time: 0.02 seconds.
-- Epoch 9
Norm: 0.07, NNZs: 3, Bias: 0.000002, T: 1800, Avg. loss: 0.664338
Total training time: 0.03 seconds.
-- Epoch 10
Norm: 0.08, NNZs: 3, Bias: 0.000002, T: 2000, Avg. loss: 0.661104
Total training time: 0.03 seconds.
-- Epoch 11
Norm: 0.09, NNZs: 3, Bias: 0.000002, T: 2200, Avg. loss: 0.657903
Total training time: 0.03 seconds.
-- Epoch 12
Norm: 0.10, NNZs: 3, Bias: 0.000001, T: 2400, Avg. loss: 0.654734
Total training time: 0.03 seconds.
-- Epoch 13
Norm: 0.11, NNZs: 3, Bias: 0.000001, T: 2600, Avg. loss: 0.651598
Total training time: 0.03 seconds.
-- Epoch 14
Norm: 0.11, NNZs: 3, Bias: 0.000001, T: 2800, Avg. loss: 0.648493
Total training time: 0.03 seconds.
-- Epoch 15
Norm: 0.12, NNZs: 3, Bias: 0.000001, T: 3000, Avg. loss: 0.645419
Total training time: 0.03 seconds.
-- Epoch 16
Norm: 0.13, NNZs: 3, Bias: 0.000002, T: 3200, Avg. loss: 0.642377
Total training time: 0.03 seconds.
-- Epoch 17
Norm: 0.14, NNZs: 3, Bias: 0.000002, T: 3400, Avg. loss: 0.639365
Total training time: 0.04 seconds.
-- Epoch 18
Norm: 0.14, NNZs: 3, Bias: 0.000001, T: 3600, Avg. loss: 0.636383
Total training time: 0.04 seconds.
-- Epoch 19
Norm: 0.15, NNZs: 3, Bias: 0.000001, T: 3800, Avg. loss: 0.633431
Total training time: 0.04 seconds.
-- Epoch 20
```

```
clf_SVM_std = linear_model.SGDClassifier(eta0=0.0001, alpha=0.0001, loss='hinge', random_s
clf_SVM_std.fit(X_std,Y)
print(clf_SVM_std.coef_)
print(clf_SVM_std.intercept_)
```

```
-- Epoch 1
Norm: 0.02, NNZs: 3, Bias: 0.000000, T: 200, Avg. loss: 0.993111
Total training time: 0.00 seconds.
-- Epoch 2
Norm: 0.03, NNZs: 3, Bias: -0.000000, T: 400, Avg. loss: 0.978934
Total training time: 0.00 seconds.
-- Epoch 3
Norm: 0.05, NNZs: 3, Bias: 0.000000, T: 600, Avg. loss: 0.964757
Total training time: 0.00 seconds.
-- Epoch 4
Norm: 0.07, NNZs: 3, Bias: 0.000000, T: 800, Avg. loss: 0.950580
Total training time: 0.01 seconds.
```

```
-- Epoch 5
Norm: 0.08, NNZs: 3, Bias: -0.000000, T: 1000, Avg. loss: 0.936403
Total training time: 0.01 seconds.
-- Epoch 6
Norm: 0.10, NNZs: 3, Bias: 0.000000, T: 1200, Avg. loss: 0.922226
Total training time: 0.01 seconds.
-- Epoch 7
Norm: 0.12, NNZs: 3, Bias: 0.000000, T: 1400, Avg. loss: 0.908049
Total training time: 0.01 seconds.
-- Epoch 8
Norm: 0.13, NNZs: 3, Bias: 0.000000, T: 1600, Avg. loss: 0.893873
Total training time: 0.01 seconds.
-- Epoch 9
Norm: 0.15, NNZs: 3, Bias: 0.000000, T: 1800, Avg. loss: 0.879696
Total training time: 0.01 seconds.
-- Epoch 10
Norm: 0.17, NNZs: 3, Bias: 0.000000, T: 2000, Avg. loss: 0.865519
Total training time: 0.01 seconds.
-- Epoch 11
Norm: 0.19, NNZs: 3, Bias: -0.000000, T: 2200, Avg. loss: 0.851342
Total training time: 0.01 seconds.
-- Epoch 12
Norm: 0.20, NNZs: 3, Bias: -0.000000, T: 2400, Avg. loss: 0.837165
Total training time: 0.01 seconds.
-- Epoch 13
Norm: 0.22, NNZs: 3, Bias: -0.000000, T: 2600, Avg. loss: 0.822988
Total training time: 0.01 seconds.
-- Epoch 14
Norm: 0.24, NNZs: 3, Bias: -0.000000, T: 2800, Avg. loss: 0.808812
Total training time: 0.01 seconds.
-- Epoch 15
Norm: 0.25, NNZs: 3, Bias: 0.000000, T: 3000, Avg. loss: 0.794635
Total training time: 0.01 seconds.
-- Epoch 16
Norm: 0.27, NNZs: 3, Bias: 0.000000, T: 3200, Avg. loss: 0.780458
Total training time: 0.01 seconds.
-- Epoch 17
Norm: 0.29, NNZs: 3, Bias: -0.000000, T: 3400, Avg. loss: 0.766282
Total training time: 0.01 seconds.
-- Epoch 18
Norm: 0.30, NNZs: 3, Bias: 0.000000, T: 3600, Avg. loss: 0.752105
Total training time: 0.01 seconds.
-- Epoch 19
Norm: 0.32, NNZs: 3, Bias: 0.000000, T: 3800, Avg. loss: 0.737929
Total training time: 0.01 seconds.
-- Epoch 20
```

```
from tabulate import tabulate
summary_lst = []
summary_lst.append(["Logistic Regression", "No standardization", clf_lr_nstd.coef_[0][0],
summary_lst.append(["Logistic Regression", "Standardization", clf_LR_std.coef_[0][0], clf_
summary_lst.append(["SVM", "No standardization", clf_SVM_Nstd.coef_[0][0], clf_SVM_Nstd.co
summary_lst.append(["SVM", "Standardization", clf_SVM_std.coef_[0][0], clf_SVM_std.coef_[0
print(tabulate(summary_lst, headers=['Classification Type', 'Data Standardization status',
```

```
Classification Type    Data Standardization status      feature1_weight_factor    fea
---------------------  -----------------------------  ------------------------  -----
Logistic Regression    No standardization                              0.371705
Logistic Regression    Standardization                                0.0385124
```

SVM	No standardization	0.382491
SVM	Standardization	0.042489

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