```
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 sklearn.linear_model import SGDClassifier
from plotly.offline import download_plotlyjs, init_notebook_mode,
plot, iplot
init_notebook_mode(connected=True)

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

```
#Task1:(a) Apply Logistic regression(SGDClassifier with logloss) on
'data' and check the feature importance


clf = SGDClassifier(loss='log',random_state =10)
clf.fit(X,Y)
print("*****************Feature Importance***************")
print("Feature Importance is given by the class weights in Logistic
regression ")
print("           class weights")
print("Feature1: ",clf.coef_[0][0])
print("Feature2: ",clf.coef_[0][1])
print("Feature3: ",clf.coef_[0][2])
```

```
*****************Feature Importance***************
Feature Importance is given by the class weights in Logistic
regression
           class weights
Feature1:  8517.365209076988
Feature2:  1947.835534937999
Feature3:  11262.189536130332
```

```
#Task1:(b) Apply SVM(SGDClassifier with hinge) on 'data' and check the
feature importance


clf = SGDClassifier(loss='hinge',random_state =10)
clf.fit(X,Y)
print("*****************Feature Importance***************")
print("Feature Importance is given by the class weights in SVM
algorithm ")
print("           class weights")
print("Feature1: ",clf.coef_[0][0])
print("Feature2: ",clf.coef_[0][1])
print("Feature3: ",clf.coef_[0][2])
```

```
*****************Feature Importance***************
Feature Importance is given by the class weights in SVM algorithm
           class weights
Feature1:  8103.543183990879
Feature2:  5998.596045031393
Feature3:  10248.44014790748
```

Observation

Explain how feature importance is affected by the correlation and variance(std-dev) of each of the features in each of the tasks. Why has one feature got more weight than the other?

- Here Feature 3 has highest correlation coeff ("0.839060") with Y. Therefore, relation between X and Y is almost linear as coef is close with 1 which infer feature 3 will be the best metric to define Y hence it is given the highest importance.
- corr(F3,Y)>corr(F1,Y)>corr(F2,Y). hence feature class-weights found likely to the corr relationship. i.e., F3>F1>F2
- var(F3)<var(F1)<var(F2) with out standardization.Feature class-weights found inversly to the var relationship. i.e., F3>F1>F2. This relation between variance and class-weight is contrast to theoritical concept. In general, more feature importance is given to the feature with large variance. This contrast is due to the scales of individual features.

Compare the results of both the models as well in each task separately and justify the difference if any

- Though the magnitude of class weights of both models found to be different but relation depicted by both model via class weights is same. i.e., F3>F1>F2.
- Relatively, logistic regression's class weights found to be more depicting the feature importance as magnitude are reflecting the significant difference for one to make proper decision on feature importance.

Conclusion: From the above results, Feature3 has more importance in classifying data points.

```
#Task2: (a) 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

#StandardScaler(): performs standardization in one step
scaler = StandardScaler()
scaler.fit_transform(X)
print("**********After Standardization**********")
print("Means:", scaler.mean_)
print("Variances:", scaler.var_)

clf = SGDClassifier(loss ='log',random_state=10)
clf.fit(X, Y)
print("*****************Feature Importance***************")
print("Feature Importance is given by the class weights in Logistic
regression ")
print("           class weights")
print("Feature1: ",clf.coef_[0][0])
print("Feature2: ",clf.coef_[0][1])
```

```
print("Feature3: ",clf.coef_[0][2])


**********After Standardization**********
Means: [  10.18003124 1299.98673919    5.00183991]
Variances: [2.37142721e+05 1.07689937e+08 8.52252180e+00]
****************Feature Importance***************
Feature Importance is given by the class weights in Logistic
regression
           class weights
Feature1:  8517.365209076988
Feature2:  1947.835534937999
Feature3:  11262.189536130332

#Task2: (a) 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

#StandardScaler(): performs standardization in one step
scaler = StandardScaler()
scaler.fit_transform(X)
print("**********After Standardization**********")
print("Means:", scaler.mean_)
print("Variances:", scaler.var_)

clf = SGDClassifier(loss ='hinge',random_state =10)
clf.fit(X, Y)
print("****************Feature Importance***************")
print("Feature Importance is given by the class weights in SVM
algorithm ")
print("           class weights")
print("Feature1: ",clf.coef_[0][0])
print("Feature2: ",clf.coef_[0][1])
print("Feature3: ",clf.coef_[0][2])

**********After Standardization**********
Means: [  10.18003124 1299.98673919    5.00183991]
Variances: [2.37142721e+05 1.07689937e+08 8.52252180e+00]
****************Feature Importance***************
Feature Importance is given by the class weights in SVM algorithm
           class weights
Feature1:  8103.543183990879
Feature2:  5998.596045031393
Feature3:  10248.44014790748
```

Observation

Explain how feature importance is affected by the correlation and variance(std-dev) of each of the features in each of the tasks. Why has one feature got more weight than the other?

- Here Feature 3 has highest correlation coeff ("0.839060") with Y. Therefore, relation between X and Y is almost linear as coef is close with 1 which infer feature 3 will be the best metric to define Y hence it is given the highest importance.
- corr(F3,Y)>corr(F1,Y)>corr(F2,Y). hence feature class-weights found likely to the corr relationship. i.e., F3>F1>F2
- var(F3)>var(F1)>var(F2) after standardization. hence feature class-weights found likely to the var relationship. i.e., F3>F1>F2. Therefore standardization nullified the effect of different scales and we found variance of features to be more important in deciding the feature importance

How did standardization impact the feature importance of different features and compare the results of this case with the non-standardized ones and explain the reason for differences between the weight vectors in both the cases.

- Standardization nullified the effect of different scales hence variances of features are intact with their feature importance
- No differences found between the weight vectors in both cases i.e., with/with out standardization.

Compare the results of both the models as well in each task separately and justify the difference if any

- Though the magnitude of class weights of both models found to be different but relation depicted by both model via class weights is same. i.e., F3>F1>F2.
- Relatively, logistic regression's class weights found to be more depicting the feature importance as magnitude are reflecting the significant difference for one to make proper decision on feature importance.

Conclusion:From the above results, Feature3 has more importance in classifying data points.Sandardizing data before applying classifier nullified the effect of scale on features and we found with variance and correlation coef to be intact with the feature importance.