IMPORTING LIBRARY

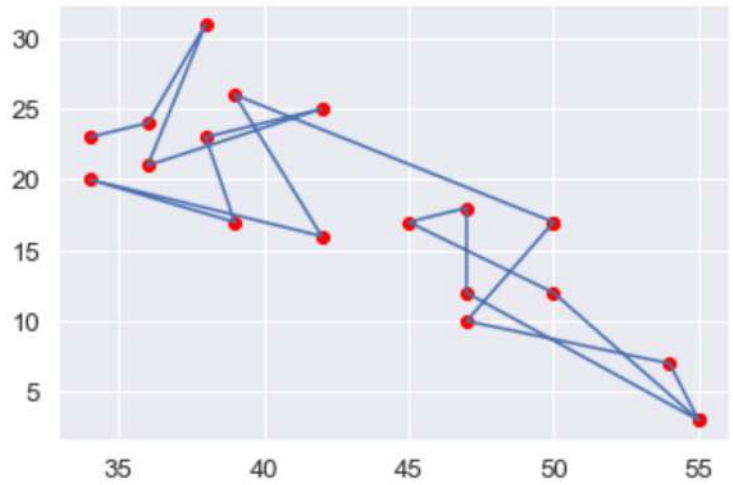
In [71]:
```
import pandas as pd
import numpy as np
from sklearn import svm
import matplotlib.pyplot as plt
import seaborn as sns; sns.set(font_scale=1.2)
%matplotlib inline
```

In [5]:
```
recipes = pd.read_csv("C:/Users\\adarsh\\Downloads\\muffin-cupcake-master\\recipes_muffins_cupcakes.csv")
recipes.head()
```

Out[5]:

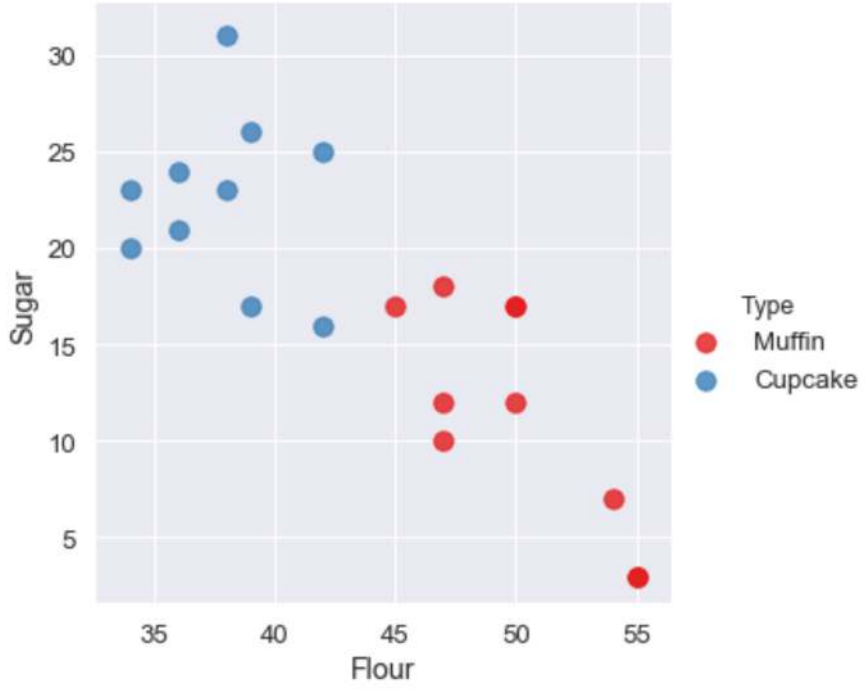
	Type	Flour	Milk	Sugar	Butter	Egg	Baking Powder	Vanilla	Salt
0	Muffin	55	28	3	7	5	2	0	0
1	Muffin	47	24	12	6	9	1	0	0
2	Muffin	47	23	18	6	4	1	0	0
3	Muffin	45	11	17	17	8	1	0	0
4	Muffin	50	25	12	6	5	2	1	0

In [32]:
```
# Plot two ingredients
import matplotlib
f=recipes['Flour']
S=recipes['Sugar']
plt.scatter(f,S,color='red')
plt.plot(f,S)
plt.show()
```



Plot two Ingredients

In [47]:
```
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type',
           palette='Set1', fit_reg=False, scatter_kws={"s": 90});
```



In [45]:
```
# Specify inputs for the model
ingredients = recipes[['Flour','Sugar']].as_matrix()
type_label = np.where(recipes['Type']=='Muffin', 0, 1)

# Feature names
recipe_features = recipes.columns.values[1:].tolist()
recipe_features
```

```
C:\Users\adarsh\Anaconda3\lib\site-packages\ipykernel_launcher.py:
2: FutureWarning: Method .as_matrix will be removed in a future ve
rsion. Use .values instead.
```

Out[45]:
```
['Flour', 'Milk', 'Sugar', 'Butter', 'Egg', 'Baking Powder', 'Vani
lla', 'Salt']
```

Fit the Model

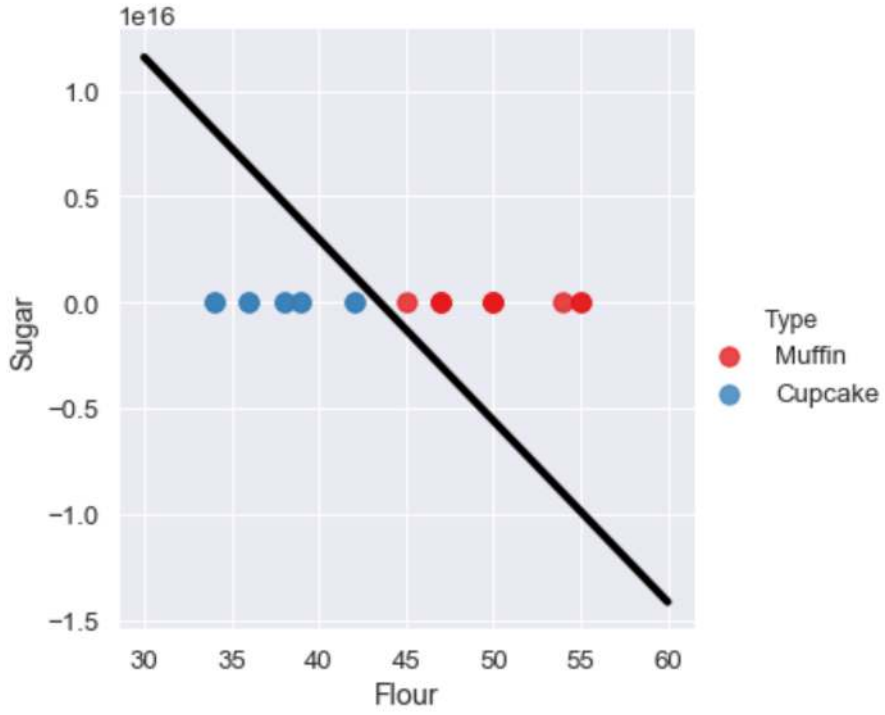
In [46]:
```
# Fit the SVM model
model = svm.SVC(kernel='linear')
model.fit(ingredients, type_label)
```

Out[46]:
```
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='auto_deprecate
d',
    kernel='linear', max_iter=-1, probability=False, random_state=
None,
    shrinking=True, tol=0.001, verbose=False)
```

In [68]:
```
# Get the separating hyperplane
w = model.coef_[0]
a = -w[0] / w[1]
xx = np.linspace(30, 60)
yy = a * xx - (model.intercept_[0]) / w[1]
```

Plot the Hyperplane

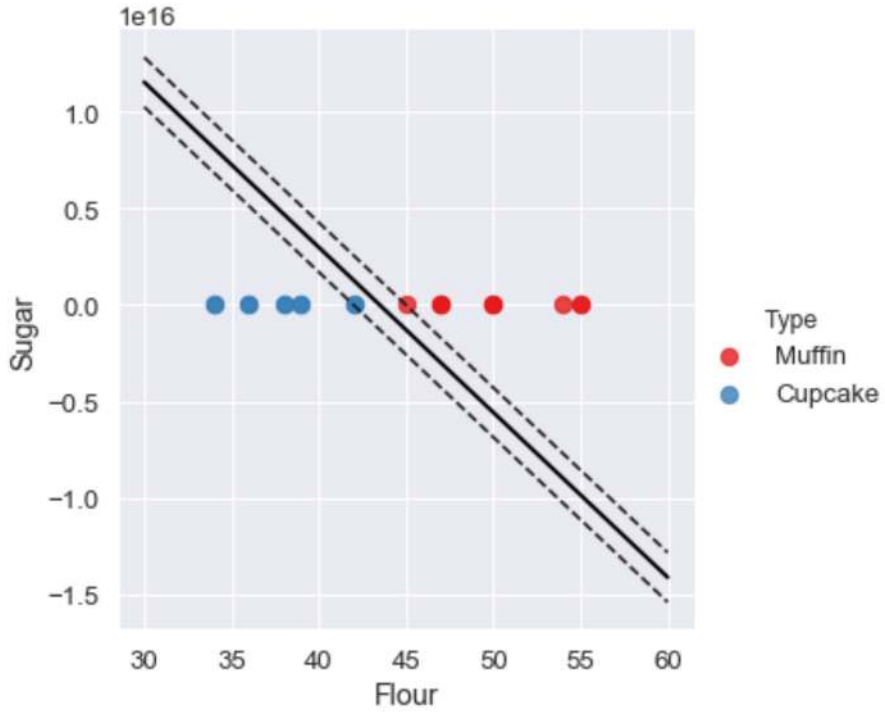
In [69]:
```
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1
', fit_reg=False, scatter_kws={"s": 90})
plt.plot(xx, yy, linewidth=4, color='black');
```



margins and support vectors

In [70]:
```
# Plot the parallels to the separating hyperplane that pass through
the support vectors
b = model.support_vectors_[0]
yy_down = a * xx + (b[1] - a * b[0])
b = model.support_vectors_[-1]
yy_up = a * xx + (b[1] - a * b[0])

# Look at the margins and support vectors
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1
', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(xx, yy_down, 'k--')
plt.plot(xx, yy_up, 'k--')
plt.scatter(model.support_vectors_[:, 0], model.support_vectors_[:,
1],
            s=80, facecolors='none');
```

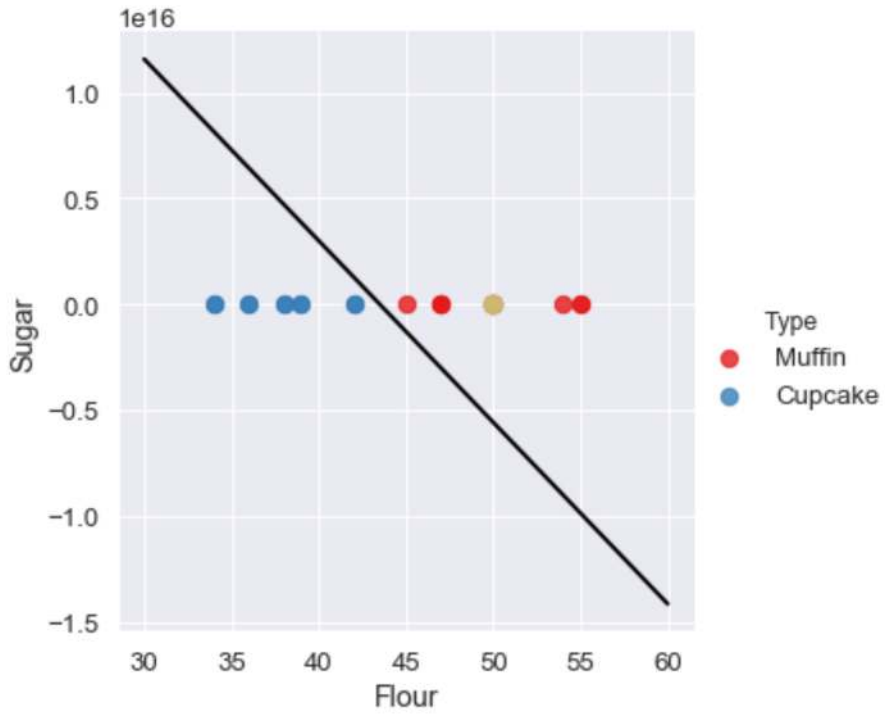


In [62]:
```
# Create a function to guess when a recipe is a muffin or a cupcake
def muffin_or_cupcake(flour, sugar):
    if(model.predict([[flour, sugar]]))==0:
        print('You\'re looking at a muffin recipe!')
    else:
        print('You\'re looking at a cupcake recipe!')
```

In [63]:
```
# Predict if 50 parts flour and 20 parts sugar
muffin_or_cupcake(50, 20)
```

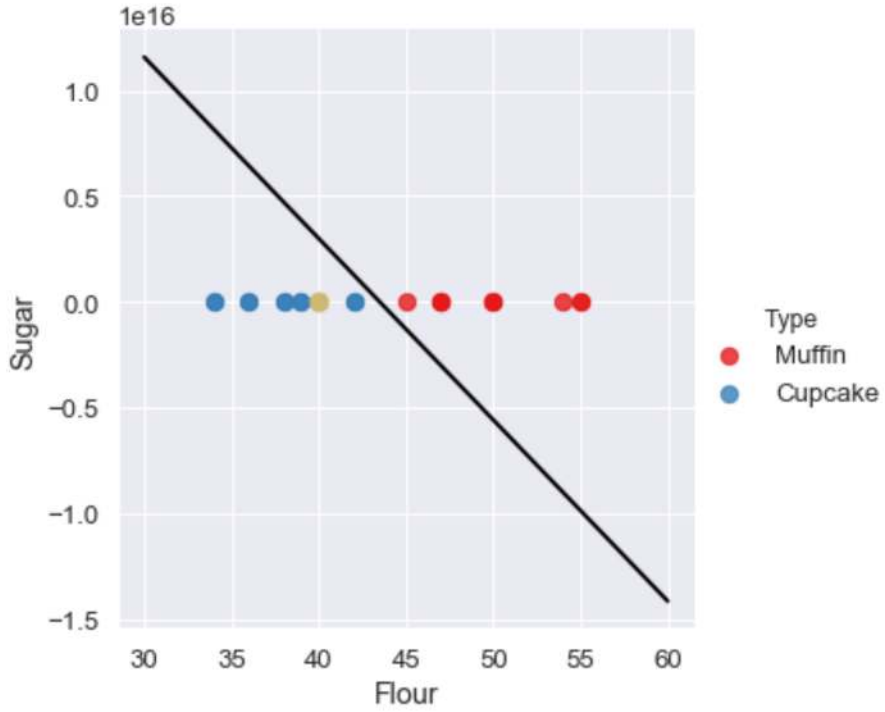
```
You're looking at a muffin recipe!
```

In [64]:
```
# Plot the point to visually see where the point lies
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1
', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(50, 20, 'yo', markersize='9');
```



In [67]:
```
# Predict if 50 parts flour and 20 parts sugar
muffin_or_cupcake(40, 20)
# Plot the point to visually see where the point lies
sns.lmplot('Flour', 'Sugar', data=recipes, hue='Type', palette='Set1
', fit_reg=False, scatter_kws={"s": 70})
plt.plot(xx, yy, linewidth=2, color='black')
plt.plot(40, 20, 'yo', markersize='9');
```

```
You're looking at a cupcake recipe!
```



In [ ]: