# HW4_P3_Jackson_Liam

May 7, 2021

**HW4 Problem 4**

### 0.1 Name: Liam Jackson

## 1 Training section

#### 1.0.1 Training imports

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import SGDClassifier, Perceptron
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline
from sklearn.metrics import confusion_matrix
import pickle
```

[2]:
```
train_df = pd.read_csv('training_dataset.csv')
train_df.drop(['id', 'Unnamed: 0'],axis=1,inplace = True)
train_df.diagnosis.replace({'M':1,'B':0}, inplace = True)
```

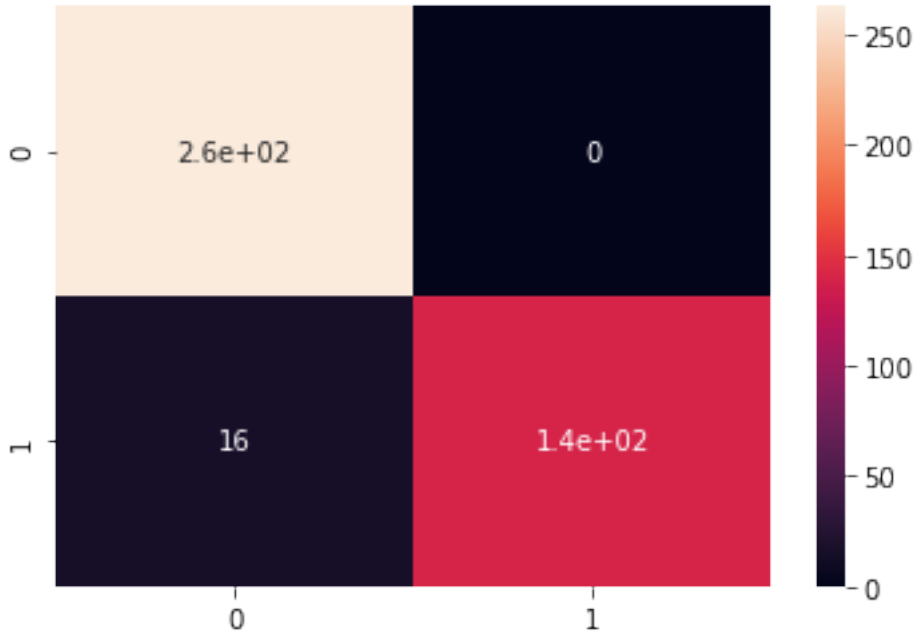
[3]:
```
X_train = train_df.drop(['diagnosis'],1).to_numpy()
y_train = train_df['diagnosis'].to_numpy()
```

#### 1.0.2 Performance and confusion matrix

[4]:
```
def knn_it(X,y,k):
    knn_model_ = KNeighborsClassifier(k, metric='canberra')
    knn_model_.fit(X,y)
    return knn_model_

nn_range = range(2,21)
k_acc = np.zeros([len(nn_range), 2])
for idx, k in enumerate(nn_range):
    knn_model = knn_it(X_train,y_train,k)
    k_vs_acc = np.array([[k, knn_model.score(X_train,y_train)]])
```

```
    k_acc[idx,:] = k_vs_acc

opt_knn_k = int(np.squeeze(k_acc[k_acc[:,1] == max(k_acc[:,1]), 0]))

knn_best = knn_it(X_train, y_train, opt_knn_k)
y_train_knn_pred = knn_best.predict(X_train)
knn_train_conf_mtx = confusion_matrix(y_train, y_train_knn_pred)
tn,fp,fn,tp = knn_train_conf_mtx.ravel()
train_sens = tp/(tp+fn)
train_spec = tn/(tn+fp)

print(f'Train set sensitivity = {train_sens}, and specificity = {train_spec}␣
 ↪for (k=3)NN model')

sns.heatmap(knn_train_conf_mtx, annot=True)

with open('knn_model_file.dat','wb') as knn_model_file:
    pickle.dump(knn_best, knn_model_file)
```

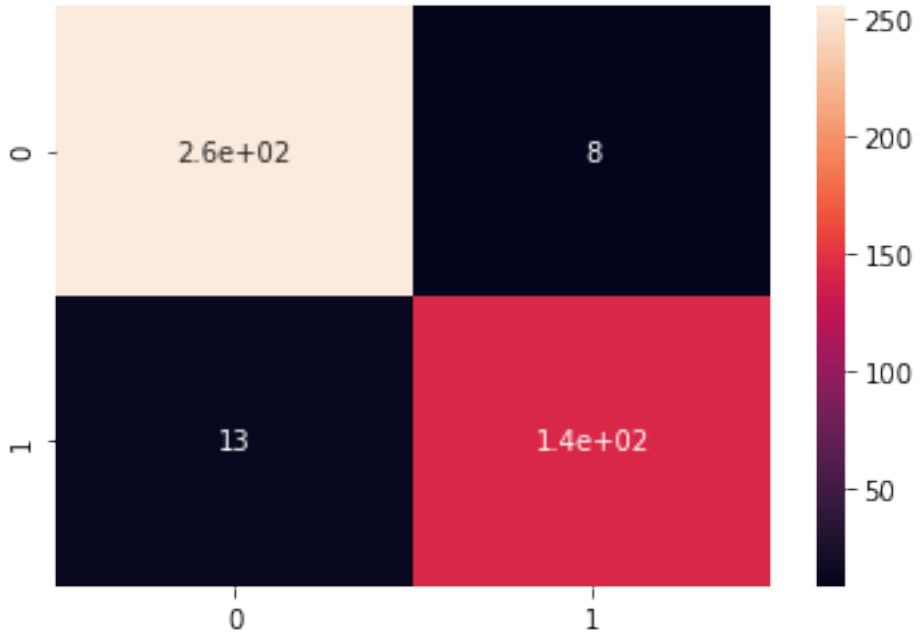
```
Train set sensitivity = 0.8974358974358975, and specificity = 1.0 for (k=3)NN
model
```




[5]:
```
sgd_model = make_pipeline(StandardScaler(),
                          SGDClassifier(loss='log',max_iter=1000, tol=1e-3,␣
 ↪random_state = 2))
sgd_model.fit(X_train,y_train)
```

```
y_train_sgd_pred = sgd_model.predict(X_train)
sgd_train_conf_mtx = confusion_matrix(y_train, y_train_sgd_pred)
tn,fp,fn,tp = sgd_train_conf_mtx.ravel()
train_sens = tp/(tp+fn)
train_spec = tn/(tn+fp)

print(f'Train set sensitivity = {train_sens}, and specificity = {train_spec}␣
 ↪for SGD Log model')

sns.heatmap(sgd_train_conf_mtx, annot=True)

with open('sgd_model_file.dat','wb') as sgd_model_file:
    pickle.dump(sgd_model, sgd_model_file)
```

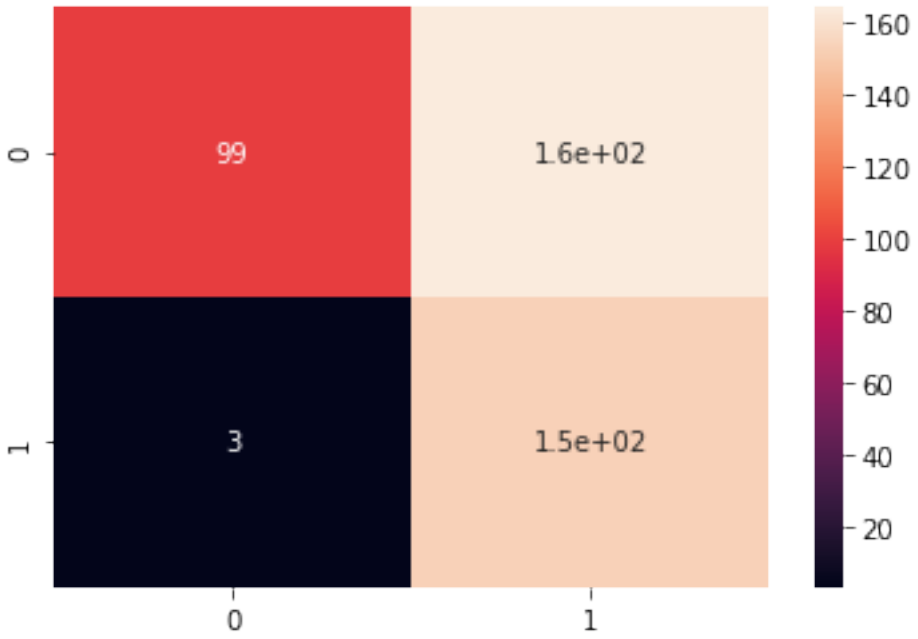
```
Train set sensitivity = 0.9166666666666666, and specificity = 0.9695817490494296
for SGD Log model
```




[6]:
```
perc_model = Perceptron(tol=1e-3,random_state=2)
perc_model.fit(X_train,y_train)
y_train_perc_pred = perc_model.predict(X_train)
perc_train_conf_mtx = confusion_matrix(y_train, y_train_perc_pred)
tn,fp,fn,tp = perc_train_conf_mtx.ravel()
train_sens = tp/(tp+fn)
train_spec = tn/(tn+fp)
```

```
print(f'Train set sensitivity = {train_sens}, and specificity = {train_spec}␣
 ↪for Perceptron model')

sns.heatmap(perc_train_conf_mtx, annot=True)

with open('perc_model_file.dat','wb') as perc_model_file:
    pickle.dump(perc_model, perc_model_file)
```

```
Train set sensitivity = 0.9807692307692307, and specificity = 0.376425855513308
for Perceptron model
```

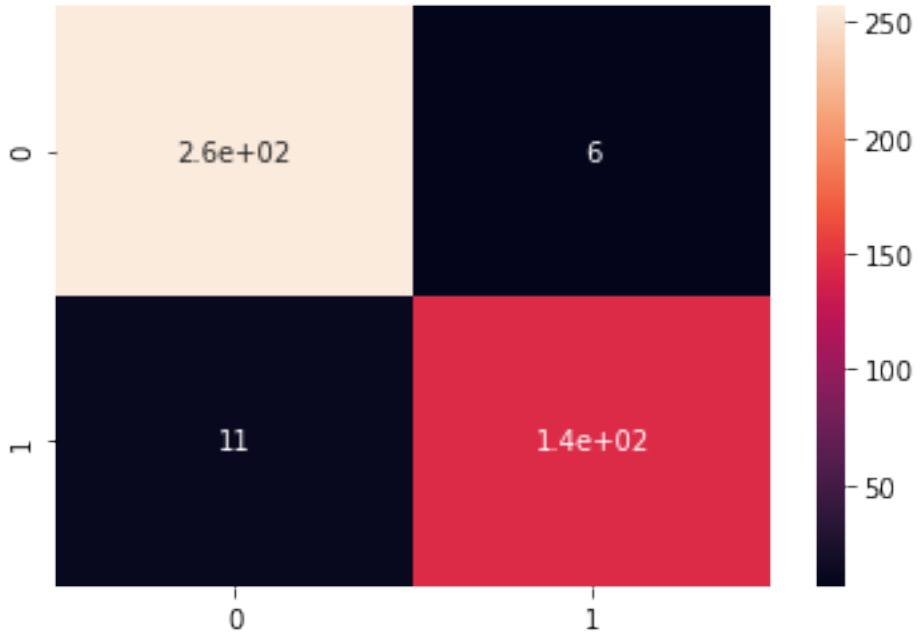



[7]:
```
train_preds_all_models = np.concatenate((y_train_knn_pred.reshape(-1,1),␣
 ↪y_train_perc_pred.reshape(-1,1), y_train_sgd_pred.reshape(-1,1)), axis=1)
train_preds_all_df = pd.DataFrame(train_preds_all_models, columns = ['knn',␣
 ↪'perc','sgd'])
voted_train_conf_mtx = confusion_matrix(y_train, train_preds_all_df.
 ↪mode(axis=1))

tn,fp,fn,tp = voted_train_conf_mtx.ravel()
train_sens = tp/(tp+fn)
train_spec = tn/(tn+fp)

print(f'Train set sensitivity = {train_sens}, and specificity = {train_spec}␣
 ↪for all 3 model consensus')
```

```
sns.heatmap(voted_train_conf_mtx, annot=True)
```

```
Train set sensitivity = 0.9294871794871795, and specificity = 0.9771863117870723
for all 3 model consensus
```

[7]: <AxesSubplot:>




## 2 Testing section

### 2.0.1 Testing imports

[8]:
```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegressionCV
from sklearn.metrics import confusion_matrix
import pickle

test_df = pd.read_csv('testing_dataset.csv')
test_df.drop(['id', 'Unnamed: 0'],axis=1,inplace = True)
test_df.diagnosis.replace({'M':1,'B':0}, inplace = True)

X_test = test_df.drop(['diagnosis'],1).to_numpy()
```

```
y_test = test_df['diagnosis'].to_numpy()
```

### 2.0.2 Testing

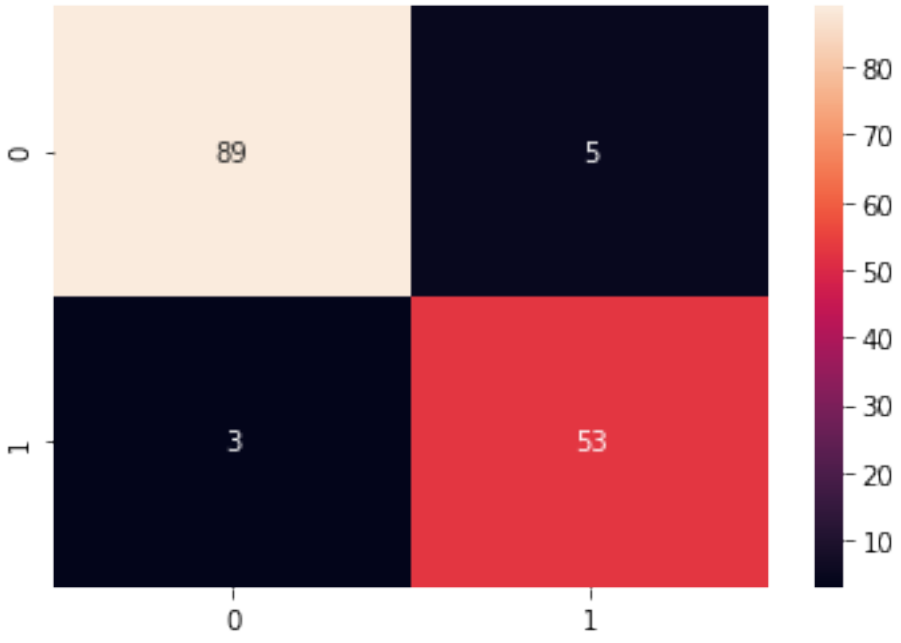
[9]:
```
with open('knn_model_file.dat','rb') as knn_model_file:
    knn_best_loaded = pickle.load(knn_model_file)

y_test_knn_pred = np.squeeze(knn_best_loaded.predict(X_test))

with open('sgd_model_file.dat','rb') as sgd_model_file:
    sgd_model_loaded = pickle.load(sgd_model_file)

y_test_sgd_pred = np.squeeze(sgd_model_loaded.predict(X_test))

with open('perc_model_file.dat','rb') as perc_model_file:
    perc_model_loaded = pickle.load(perc_model_file)

y_test_perc_pred = np.squeeze(perc_model_loaded.predict(X_test))
```

### 2.0.3 Performance and confusion matrix

[10]:
```
test_preds_all_models = np.concatenate((y_test_knn_pred.reshape(-1,1),␣
↪y_test_perc_pred.reshape(-1,1), y_test_sgd_pred.reshape(-1,1)), axis=1)
test_preds_all_df = pd.DataFrame(test_preds_all_models, columns = ['knn',␣
↪'perc','sgd'])
voted_train_conf_mtx = confusion_matrix(y_test, test_preds_all_df.mode(axis=1))

tn,fp,fn,tp = voted_train_conf_mtx.ravel()
train_sens = tp/(tp+fn)
train_spec = tn/(tn+fp)

print(f'Train set sensitivity = {train_sens}, and specificity = {train_spec}␣
↪for all 3 model consensus')

sns.heatmap(voted_train_conf_mtx, annot=True)
```

```
Train set sensitivity = 0.9464285714285714, and specificity = 0.9468085106382979
for all 3 model consensus
```

[10]:
```
<AxesSubplot:>
```


89
5
3
53
0
1
0
1
80
70
60
50
40
30
20
10