In [1]:

```
import pandas as pd
from matplotlib import pyplot as plt
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, recall_score, precision_score, f1_
score, classification_report
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.dummy import DummyClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import cross_val_score
```

In [6]:

```
data = pd.read_csv('../datasets/train.csv')
```

In [7]:

```
data.head()
```

Out[7]:

	blueFirstBlood	blueKills	blueDeaths	blueGoldDiff	blueExperienceDiff	blueWardsPlacedDiff	b
0	1	11	9	1433	508	-11	
1	0	6	4	533	1187	-2	
2	0	3	4	3156	3919	4	
3	0	2	9	-3084	-1719	-6	
4	0	3	7	-2825	-2497	-7	

In [8]:

```
X = data.loc[:, data.columns != 'blueWins']
y = data['blueWins']
```

In [9]:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rando
m_state=42)
```

In [14]:

```
pipe_dummy = make_pipeline(DummyClassifier(strategy="stratified",random_state=
23))
pipe_dummy.fit(X_train, y_train)
y_pred = pipe_dummy.predict(X_test)
print('Test accuaracy: %.3f' % pipe_dummy.score(X_test, y_test))
```

```
Test accuaracy: 0.509
```

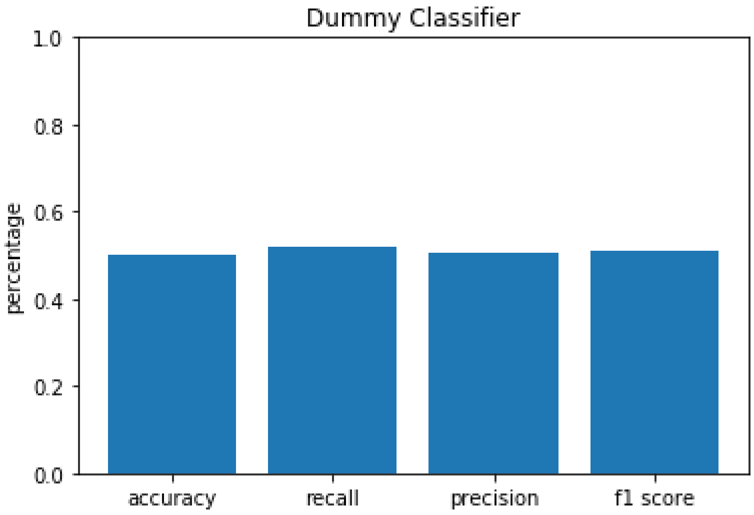
In [15]:

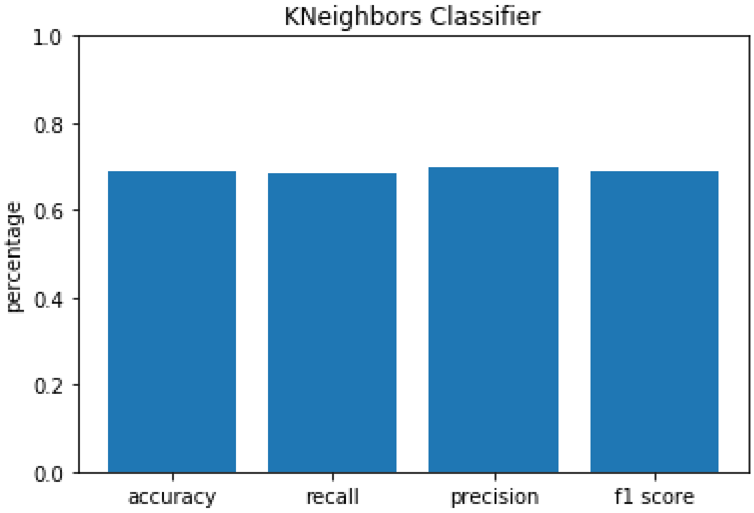
```
pipe_dummy = make_pipeline(StandardScaler(),
                        DummyClassifier(strategy="stratified",random_state=11
))
pipe_dummy.fit(X_train, y_train)
y_pred = pipe_dummy.predict(X_test)
print('Test accuaracy: %.3f' % pipe_dummy.score(X_test, y_test))
print(classification_report(y_test, y_pred))
```

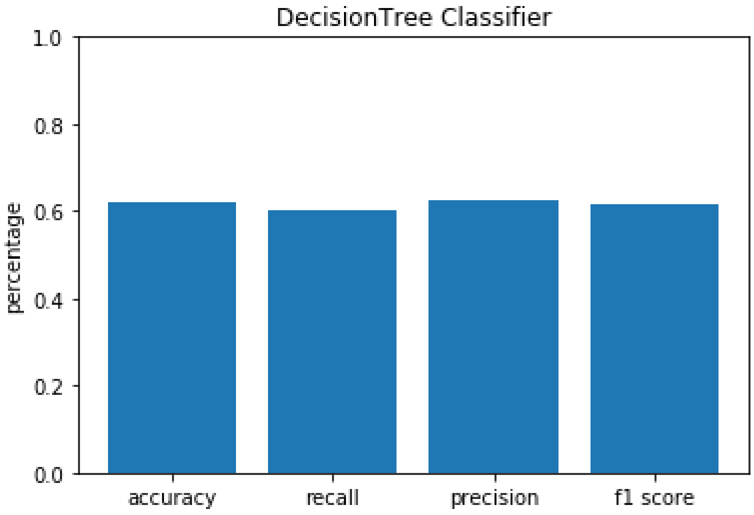
```
Test accuaracy: 0.507
              precision    recall  f1-score   support

           0       0.50      0.51      0.51      1028
           1       0.51      0.50      0.51      1047

    accuracy                           0.51      2075
   macro avg       0.51      0.51      0.51      2075
weighted avg       0.51      0.51      0.51      2075
```

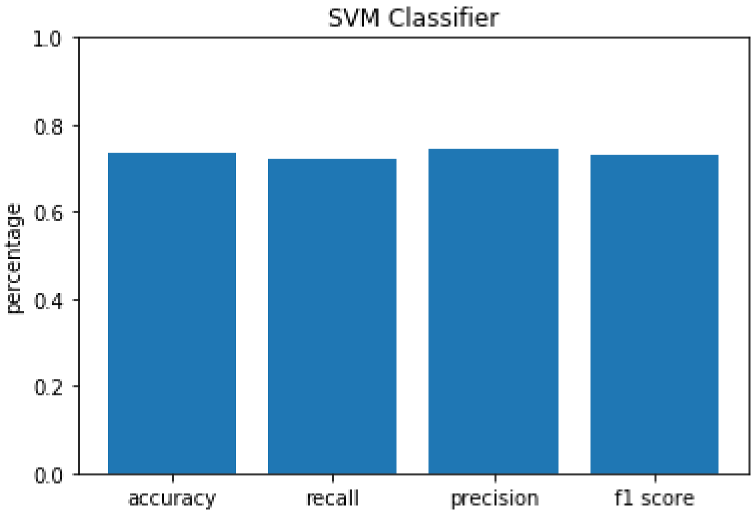
In [16]:

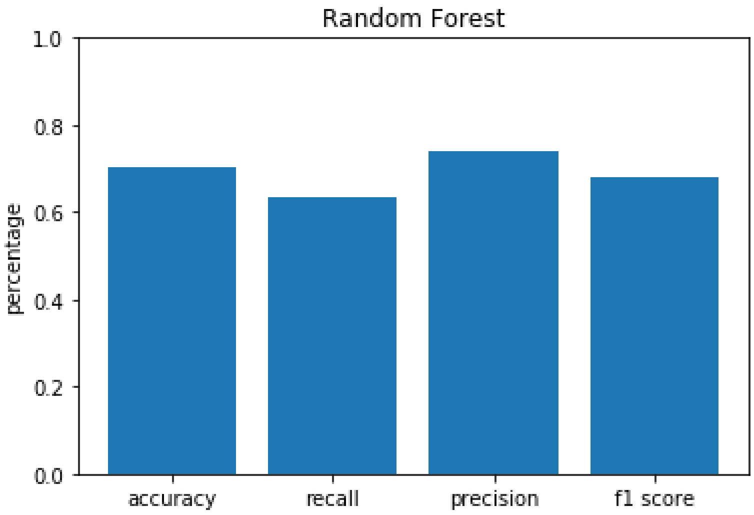
```
def Logistic_Regression():
    lr_clf = LogisticRegression(max_iter=1000,solver='lbfgs',random_state=5)
    return lr_clf
def KNeighbors_Classifier():
    knn_clf = KNeighborsClassifier()
    return knn_clf
def DecisionTree_Classifier():
    tree_clf = DecisionTreeClassifier(random_state=6)
    return tree_clf
def Random_Forest():
    rf_clf = RandomForestClassifier(n_estimators=10,random_state=1)
    return rf_clf
def Svm_Classifier():
    svm_clf = SVC(gamma='scale',random_state=3)
    return svm_clf
def graph_model(clfs):
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)
    metricArray = [accuracy_score(y_test, y_pred), recall_score(y_test, y_pred
),
                   precision_score(y_test, y_pred), f1_score(y_test, y_pred)]
    x=[0,1,2,3]
    labels=['accuracy','recall', 'precision', 'f1 score']
    plt.bar(x,metricArray)
    plt.xticks(x,labels)
    plt.ylim(0,1)
    plt.ylabel('percentage')
    plt.title(label)
    plt.show()

#“stratified”: generates predictions by respecting the training set’s class di
stribution.
dummy_clf = DummyClassifier(strategy="stratified",random_state=99)
clfs = [dummy_clf, KNeighbors_Classifier(), DecisionTree_Classifier(),
        Svm_Classifier(), Random_Forest(),Logistic_Regression()]
clf_labels = ['Dummy Classifier','KNeighbors Classifier','DecisionTree Classif
ier',
              'SVM Classifier','Random Forest','Logistic Regression']
for label,clf in zip(clf_labels,clfs):
    graph_model(clf)
```

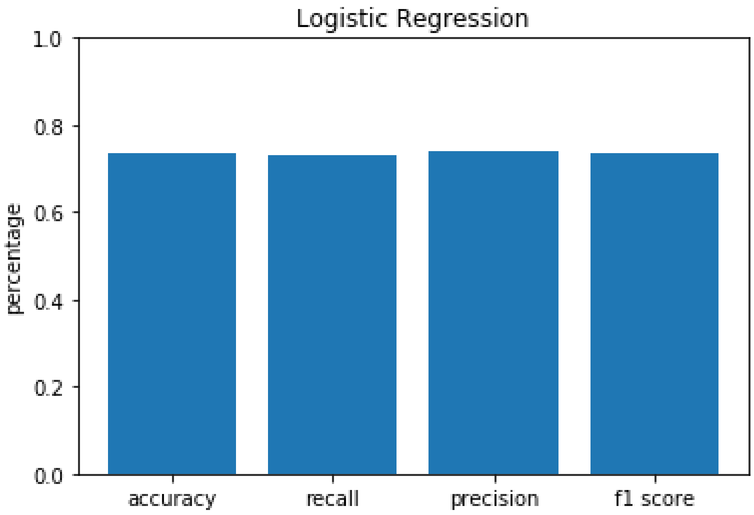

Dummy Classifier
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score


KNeighbors Classifier
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score


DecisionTree Classifier
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score


SVM Classifier
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score


Random Forest
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score


Logistic Regression
percentage
1.0
0.8
0.6
0.4
0.2
0.0
accuracy
recall
precision
f1 score

In [17]:

```
for label,clf in zip(clf_labels,clfs):
    scores = cross_val_score(estimator=clf, X=X_train, y=y_train, cv=5)
    print("accuracy: %0.3f (+/- %0.3f) [%s]"
    % (scores.mean(), scores.std(), label))
```

```
accuracy: 0.495 (+/- 0.013) [Dummy Classifier]
accuracy: 0.676 (+/- 0.004) [KNeighbors Classifier]
accuracy: 0.626 (+/- 0.018) [DecisionTree Classifier]
accuracy: 0.727 (+/- 0.007) [SVM Classifier]
accuracy: 0.693 (+/- 0.017) [Random Forest]
accuracy: 0.733 (+/- 0.014) [Logistic Regression]
```

In [ ]: