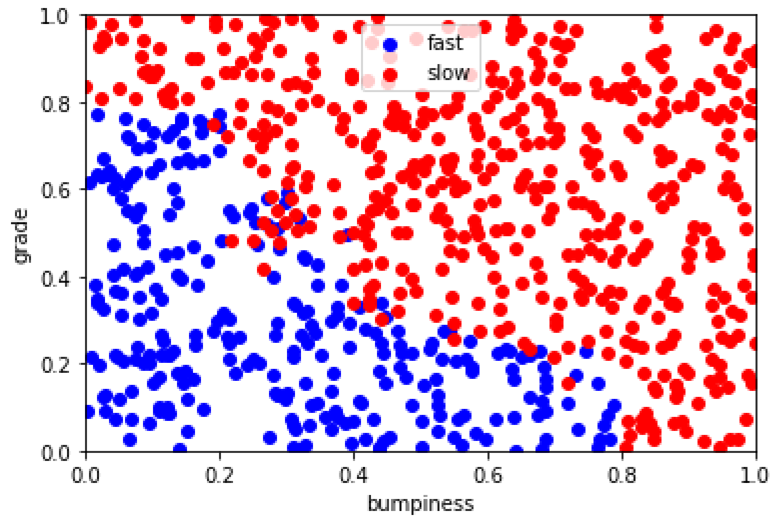
In [4]:
```
#!/usr/bin/python

import matplotlib.pyplot as plt
from prep_terrain_data import makeTerrainData
from class_vis import prettyPicture

features_train, labels_train, features_test, labels_test = makeTerrainData()


### the training data (features_train, labels_train) have both "fast" and "slow"
### points mixed together--separate them so we can give them different colors
### in the scatterplot and identify them visually
grade_fast = [features_train[ii][0] for ii in range(0, len(features_train)) if la
bumpy_fast = [features_train[ii][1] for ii in range(0, len(features_train)) if la
grade_slow = [features_train[ii][0] for ii in range(0, len(features_train)) if la
bumpy_slow = [features_train[ii][1] for ii in range(0, len(features_train)) if la


#### initial visualization
plt.xlim(0.0, 1.0)
plt.ylim(0.0, 1.0)
plt.scatter(bumpy_fast, grade_fast, color = "b", label="fast")
plt.scatter(grade_slow, bumpy_slow, color = "r", label="slow")
plt.legend()
plt.xlabel("bumpiness")
plt.ylabel("grade")
plt.show()
################################################################################
```



A critical skill for any data analyst is the ability to figure out new things about machine learning, which is the goal for this lesson. The whole lesson is a mini-project. The goal is to do terrain classification with an algorithm of your choice, researching and deploying it on your own.

Your algorithm choices are the following:

- k nearest neighbors

- random forest
- adaboost (sometimes also called boosted decision tree)

We can’t check your results, because there are too many combinations of algorithms and parameters to check everything that you could try, but you have seen the accuracy that our previous algorithms (Naive Bayes, SVM, decision tree) achieved and can self-assess whether the new algorithm does better.

In the choose_your_own/your_algorithm.py file, you’ll find some starter code to get the data all set up for you. The following videos also give a little more background on the algorithms and process you should follow, but you’re mostly finding your own way here. Good luck!

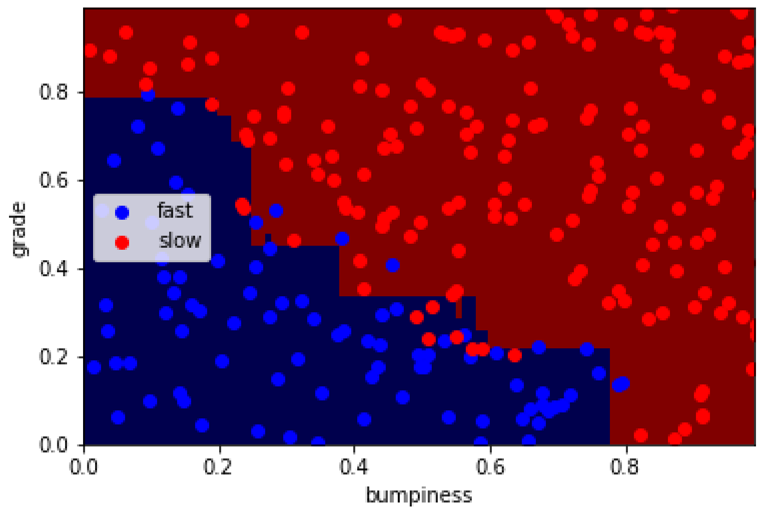
In [5]:
```
### your code here!  name your classifier object clf if you want the
### visualization code (prettyPicture) to show you the decision boundary

from sklearn.ensemble import AdaBoostClassifier as adc
clf = adc()
clf.fit(features_train,labels_train)
pred = clf.predict(features_test)

from sklearn.metrics import accuracy_score as sco
acc = sco(labels_test,pred)
print"AdaBoost Accuracy: ",acc

try:
    prettyPicture(clf, features_test, labels_test)
except NameError:
    pass
```

```
AdaBoost Accuracy:  0.924
```



In [ ]: