Polynomial Kernel

The linear kernel function that we implemented in the previous part of this lab, is the simplest way to train an SVM model. However, we can make our SVM model more powerful by using more complicated kernel functions. In this part of the lab, we will be implementing a second-degree polynomial kernel that should produce a slightly better SVM model than the model that we obtained using a linear kernel function.

Just like before, your function should take two feature matrices `X1` and `X2` as its inputs, and compute a Kernel matrix `K`. However, in this case, every entry `K(i,j)` should store the square of the dot product between 1) the feature vector from row `i` in matrix `X1`, and 2) the feature vector from row `j` in matrix `X2`. In other words, the dot product between feature vectors should be raised to the power of `2`.

Your function should be implemented as a one line matrix multiplication without using `for` loops, which would make SVM training more computationally expensive. Your SVM model using a second deegree polynomial kernel function should achieve 89.50% accuracy on the testing dataset, which is slightly better than our previous model.

Your Function

Save Reset MATLAB Documentation (https://www.mathworks.com/help/)

```
function K = KernelPoly(X1, X2)
   % computes a second degree polynomial kernel
   %
   % Input:
   % - X1: an n x d dimensional feature matrix where n is the number of observations, and d is the number of fe
   % - X2: an m x d dimensional feature matrix where m is the number of observations, and d is the number of fe
   % Output:
   % - K: an n x m dimensional kernel matrix where K(i,j) stores a square of the dot product between the data p
   K = X1 * X2';
   K = K.^2;
end

```

Code to call your function

Reset

```
load('ImageDataTrain.mat');
Xtrain=StandardizeData(data.trainX);
Ytrain=data.trainY;

load('ImageDataTest.mat');
Xtest=StandardizeData(data.testX);
Ytest=data.testY;

model = fitcsvm(Xtrain,Ytrain,'KernelFunction','KernelPoly');
[preds,~] = predict(model,Xtest);
```

▶ Run Function

Previous Assessment: All Tests Passed

Submit

- ✓ **Is the Polynomial Kernel Function Correct?**
- ✓ **Does the Solution Exclude Built in 'for' Loops?**