Top 6,4, .97

1. Education gain .1245 vs .1245 for Career and .0202 for Experience
    1. 1. High School 4,1, Entropy = .722
        1. Experience Gain: .322 > Career Gain of 0.171
            1. Less than 3: 1,0 Entropy = 0, all low
            2. 3 to 10: 2,0 Entropy = 0, all low
            3. More than 10, 1,1 Entropy = 1
        1. Mgmt 0,1 Entropy = 0, all high
        2. Service 1,0 Entropy = 0, all low
    2. College 2,3 Entropy = .971
        1. Career Gain: .4202 > Experience Gain of .171

1. Mgmt 0,2 Entropy = 0, all high
2. Service 2,1 Entropy = 0.918

Experience Gain .918

Less than 3: 1,0 Entropy = 0, all low

3. 3 to 10: 0,1 Entropy = 0, all high

More than 10, 1,0 Entropy = 0, all low

Step 2: Prune the Tree Using Validation Set

Following Above Tree for Each Instance

Instance 1:

High School → More than 10 → Management → High, Matches Instance's High Salary

Do not prune the High school tree bc there are no validation errors.

Instance 2:

College → Mgmt → High: Does not match Instance's low salary

2 training obs are correct and this 1 testing obs is wrong, keep as is.

Instance 3:

College → Service → 3 to 10 → High, Does not match Instance's low salary

Prune this so that all College → Service are predicted as Low salary.

Doesn't make sense to predict College -> service as high just because of this instance.

Final Pruned Tree and # Correct Using all 13 Instances (Training and Test)

1. HS
    2. Experience less than 10 years → Low Correct: 3 Errors: 0
    2. Experience More than 10 Years
        3. Management → High Correct: 2 Errors: 0
        3. Service → Low Correct: 1 Errors: 0

1. College
    2. Management → High Correct: 2 Errors: 1
    2. Service → Low Correct: 3 Errors: 1

2. PROGRAM KNN in Python, Code Attached

ReadMe
I was unable to get an arff reader to work on either my mac or at the CS dept computers. Therefore I uploaded the data
as a csv and hardcoded the np arrays after reading them in.

To calculate EucDist between two points, use:
EucDist(a,b,length) where length specfies how many numerical variables there are
to calculate distance, assuming all non-numeric variables exist after the numbers.
eg for a= [6.5, 3. , 5.8, 2.2], and b = [5.1, 3.3, 1.7, 0.5]

EucDist(a,b,4)

To find the K-NN of a point, enter in the training set, the test observation, and k to test using:
NearestNeighbors(train, testobs, k)

To Predict the class of an item using the test observation and the list of Nearest neighbors use:
PredictClass(test,neighbors)

To retrieve the results for this assignment enter either:
results
or
addColsk(test,train)

To perform the same analysis on a different dataset, enter in the training array 'train' and
the testing array test into addColsk(test,train). Or for a single obs:
enter
newobs = np.array([[6.7, 3.1, 4.4, 1.4]])
and then test using the same training set:
addColsk(newobs, train)

returning: array([['6.7', '3.1', '4.4', '1.4', 'x0', 'versicolor', 'versicolor',
'versicolor', 'versicolor', 'versicolor']],
dtype='|S32')

3. WEKA
10 Fold CV SMO Classification Results

Run 1: PolyKernel with exponent option 1 # Correctly Classified: 717 # Incorrect: 129
Run 2: PolyKernel with exponent option 2 # Correctly Classified: 810 # Incorrect: 36
Run 3: PolyKernel with exponent option 4 # Correctly Classified: 781 # Incorrect: 55
Run 4: RBFKernel with Gamma set to 0.01 # Correctly Classified: 614 # Incorrect: 232
Run 5: RBFKernel with Gamma set to 1.0 # Correctly Classified: 764 # Incorrect: 82

The RBFKernel with Gamma equal to 1 and the PolyKernel with exponent equal to 1 did not work well.
The PolyKernel works better with higher exponents because it allows a higher degree function to map the

data rather than restricting the data to a linear function with exponent 1. You can see that the fourth degree option did not outperform option 2 because it likely overfit the training data with such a high degree function relative to the second option. The RBFKernel gamma parameter defines how far the influence of a single observation reaches. This Kernel worked better with gamma set to 1 rather than .01 because .01 allows a training example's influence to reach farther which does not work well in this classification setting.

4. Kernels

K(x,z)=x1 ∗z1 +x1 ∗e^z2 +z1 ∗e^x2 +e^(x2+z2).

Is K(x,z) an inner product of some Φ (x)? Yes:

$\Phi(x) = \Phi(x_1,x_2) = [x_1, e^{(x_2)}]$

$\Phi(z) = \Phi(z_1,z_2) = [z_1, e^{(z_2)}]$

$K(x,z) = [x_1, e^{(x_2)}]' * [z_1, e^{(z_2)}]$ = x1 ∗z1 +x1 ∗e^z2 +z1 ∗e^x2 +e^(x2+z2).

It is possible to prove K is a kernel by showing K is symmetric and the matrix is positive semi-definite for x1,....xN.

If x = (-1,1) and z = (1,-1)

K(x,z) = 2.35, k(-x,-z) = 2.35 SYMMETRIC, K(u) = K(-u) for all u.

Satisfies Mercer's condition.

5. Dual Form SVM
   1.

w = **[2.7, -2, -1.8]**

yi(wTxi + b) = 1

Using first row: yi(wTxi + b) = 0.9 + b = 1, so **b = 0.1**

For alpha 2

Sign(wTxi + b) = (2.7*1 – 2*.8 -1.8*1) + 0.1) = sign(-0.6) = -1

S**o new observation is predicted to be a -1.**