# Group 5

Code for this assignment at https://github.com/TetroVolt/COSC3337Assign2

# Who we are

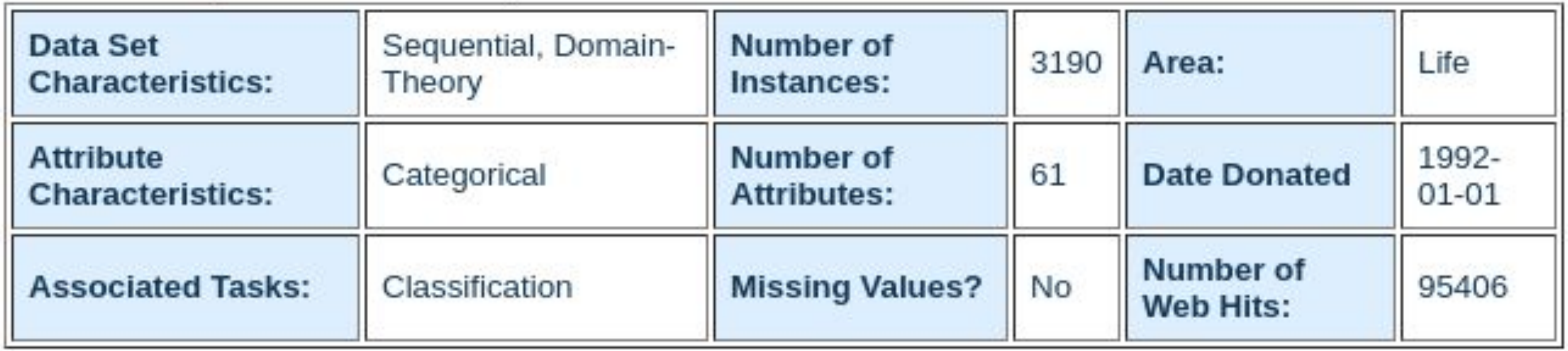
Raymond Sutrisno

Niels Moeller

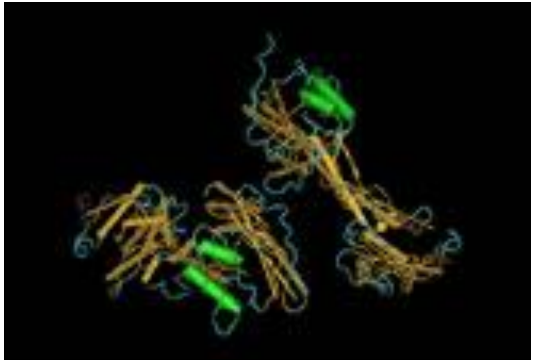
Gal Egozi

Colby Kuhnel

# Data Set I (Molecular Biology Data Set)



| | | | | | |
|---|---|---|---|---|---|
| **Data Set Characteristics:** | Sequential, Domain-Theory | **Number of Instances:** | 3190 | **Area:** | Life |
| **Attribute Characteristics:** | Categorical | **Number of Attributes:** | 61 | **Date Donated** | 1992-01-01 |
| **Associated Tasks:** | Classification | **Missing Values?** | No | **Number of Web Hits:** | 95406 |

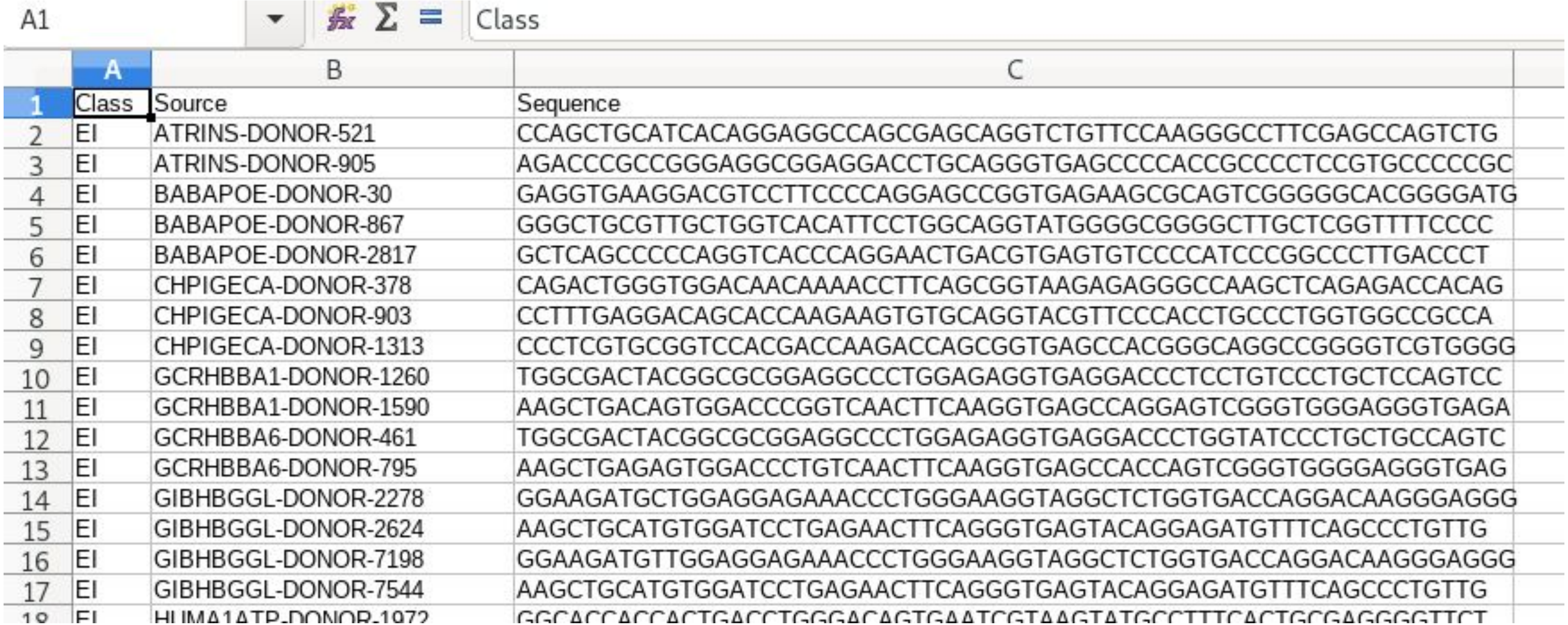


- Introns do not code for proteins
- Exons code for proteins

# Data Set II: Preliminary analysis

1. Each Sequence is 60 base pairs long
2. 8 possible base pairs: ('A', 'C', 'D', 'G', 'N', 'R', 'S', 'T')
3. Three Class Labels
    a. Exon-Intron Boundaries (EI) (767 examples)
    b. Intron-Exon Boundaries (IE) (768 examples)
    c. Neither (N) (1655 examples)

# Data Preprocessing I (RAW data)



| | A | B | C |
|---|---|---|---|
| 1 | Class | Source | Sequence |
| 2 | EI | ATRINS-DONOR-521 | CCAGCTGCATCACAGGAGGCCAGCGAGCAGGTCTGTTCCAAGGGCCTTCGAGCCAGTCTG |
| 3 | EI | ATRINS-DONOR-905 | AGACCCGCCGGGAGGCGGAGGACCTGCAGGGTGAGCCCCACCGCCCCTCCGTGCCCCCGC |
| 4 | EI | BABAPOE-DONOR-30 | GAGGTGAAGGACGTCCTTCCCCAGGAGCCGGTGAGAAGCGCAGTCGGGGGCACGGGGATG |
| 5 | EI | BABAPOE-DONOR-867 | GGGCTGCGTTGCTGGTCACATTCCTGGCAGGTATGGGGCGGGGCTTGCTCGGTTTTCCCC |
| 6 | EI | BABAPOE-DONOR-2817 | GCTCAGCCCCCAGGTCACCCAGGAACTGACGTGAGTGTCCCCATCCCGGCCCTTGACCCT |
| 7 | EI | CHPIGECA-DONOR-378 | CAGACTGGGTGGACAACAAAACCTTCAGCGGTAAGAGAGGGCCAAGCTCAGAGACCACAG |
| 8 | EI | CHPIGECA-DONOR-903 | CCTTTGAGGACAGCACCAAGAAGTGTGCAGGTACGTTCCCACCTGCCCTGGTGGCCGCCA |
| 9 | EI | CHPIGECA-DONOR-1313 | CCCTCGTGCGGTCCACGACCAAGACCAGCGGTGAGCCACGGGCAGGCCGGGGTCGTGGGG |
| 10 | EI | GCRHBBA1-DONOR-1260 | TGGCGACTACGGCGCGGAGGCCCTGGAGAGGTGAGGACCCTCCTGTCCCTGCTCCAGTCC |
| 11 | EI | GCRHBBA1-DONOR-1590 | AAGCTGACAGTGGACCCGGTCAACTTCAAGGTGAGCCAGGAGTCGGGTGGGAGGGTGAGA |
| 12 | EI | GCRHBBA6-DONOR-461 | TGGCGACTACGGCGCGGAGGCCCTGGAGAGGTGAGGACCCTGGTATCCCTGCTGCCAGTC |
| 13 | EI | GCRHBBA6-DONOR-795 | AAGCTGAGAGTGGACCCTGTCAACTTCAAGGTGAGCCACCAGTCGGGTGGGGAGGGTGAG |
| 14 | EI | GIBHBGGL-DONOR-2278 | GGAAGATGCTGGAGGAGAAACCCTGGGAAGGTAGGCTCTGGTGACCAGGACAAGGGAGGG |
| 15 | EI | GIBHBGGL-DONOR-2624 | AAGCTGCATGTGGATCCTGAGAACTTCAGGGTGAGTACAGGAGATGTTTCAGCCCTGTTG |
| 16 | EI | GIBHBGGL-DONOR-7198 | GGAAGATGTTGGAGGAGAAACCCTGGGAAGGTAGGCTCTGGTGACCAGGACAAGGGAGGG |
| 17 | EI | GIBHBGGL-DONOR-7544 | AAGCTGCATGTGGATCCTGAGAACTTCAGGGTGAGTACAGGAGATGTTTCAGCCCTGTTG |
| 18 | EI | HUMA1ATP-DONOR-1972 | GGCACCACCACTGACCTGGGACAGTGAATCGTAAGTATGCCTTTCACTGCGAGGGGTTCT |

# Algorithms considered

Required:

1. Neural Networks (Gal Egozi)
2. Support Vector Machines (Colby Kuhnel)

Chosen freely:

1. Decision Tree (Niels Moeller)
2. Random Forest (Raymond Sutrisno)

# Data Processing II (considerations)

1. NN and SVM are “One vs. All” algorithms
2. Features are categorical in nature
3. Need One Hot Encoding for both class labels and features

# Data Processing III (procedure)

1. One hot encode features
   a. Each Sequence is 60 base pairs long and categorical
   b. 60 features
   c. 8 base pair categories ('A', 'C', 'D', 'G', 'N', 'R', 'S', 'T')
   d. 60 * 8 = 480 columns total after one hot encoding
2. One hot encode classes for algorithms that need it
   a. Three classes: ‘EI’ , ‘IE’, ‘N’

# Training Procedure

1. Gridsearch model parameters to tune for the best model using stratified 10 fold cross validation
2. Report accuracies (CV mainly, training and test)
3. Optional metrics considered
    a. Confusion Matrices
    b. Learning Curves

# Implementation Info

1. Language used: Python
2. Utility Libraries:
   a. Pandas (data processing)
   b. Numpy (data processing)
   c. Scikit Learn (data processing)
3. Machine Learning Libraries
   a. Scikit Learn (SVM, Random Forest, Decision Tree)
   b. Keras (Neural networks)

# Results

# Random Forest (trees based on C4.5)

# Random Forest I (model parameter space available)

1. Criterion [‘gini’, ‘entropy’]
2. Max_depth
3. Min sample splits
4. Min impurity decrease
5. ...

# Random Forest I (model parameter space used)

```
1.  'N_estimators':[25, 50, 100, 150]
2.  'max_depth': [2, 4, 8, 16, 32, 64]
```

# Random Forest II (Raw Output) (20 % Test)

```
Dataset characteristics:
  Number of examples in the dataset = 3190
  Number of examples reserved for test set = 638
  Number of examples reserved for training via 10 fold CV = 2552
  Class Distribution Ratio (N : EI : IE) = 2 : 1 : 1
  N features : 60, all categorical (DNA base pairs in 60 base pair long sequence)


  Grid Search parameter space for Random Forest =
     {'n_estimators': [25, 50, 100, 150], 'max_depth': [2, 4, 8, 16, 32, 64]}
  best estimator parameters found = {'max_depth': 32, 'n_estimators': 150}
  best estimator mean training score   = 0.9995646116203684
  best estimator mean validation score = 0.9114420062695925
  best estimator test score          = 0.9247648902821317
```

# Random Forest III (Raw Output) (50 % Test)

```
Dataset characteristics:
  Number of examples in the dataset = 3190
  Number of examples reserved for test set = 1595
  Number of examples reserved for training via 10 fold CV = 1595
  Class Distribution Ratio (N : EI : IE) = 2 : 1 : 1
  N features : 60, all categorical (DNA base pairs in 60 base pair long sequence)


  Grid Search parameter space for Random Forest =
     {'n_estimators': [25, 50, 100, 150], 'max_depth': [2, 4, 8, 16, 32, 64]}
  best estimator parameters found = {'max_depth': 16, 'n_estimators': 100}
  best estimator mean training score   = 0.9992337891743421
  best estimator mean validation score = 0.8915360501567398
  best estimator test score           = 0.8984326018808777
```

# Random Forest IV (Raw Output) (70 % Test)

```
Dataset characteristics:
  Number of examples in the dataset = 3190
  Number of examples reserved for test set = 2233
  Number of examples reserved for training via 10 fold CV = 957
  Class Distribution Ratio (N : EI : IE) = 2 : 1 : 1
  N features : 60, all categorical (DNA base pairs in 60 base pair long sequence)


  Grid Search parameter space for Random Forest =
     {'n_estimators': [25, 50, 100, 150], 'max_depth': [2, 4, 8, 16, 32, 64]}
  best estimator parameters found = {'max_depth': 16, 'n_estimators': 100}
  best estimator mean training score   = 1.0
  best estimator mean validation score = 0.8610240334378265
  best estimator test score           = 0.8669950738916257
```

# Random Forest Best Parameters

| Train : Test Ratio | Max Depth | N Estimators |
|---|---|---|
| 80:20 | 32 | 150 (MAX) |
| 50:50 | 16 | 100 |
| 30:70 | 16 | 100 |

# Random Forest Mean Scores

| Train : Test Ratio | Mean Train | Mean CV | Test Score |
|---|---|---|---|
| 80:20 | 0.9995 | 0.9114 | 0.9247 |
| 50:50 | 0.9992 | 0.8915 | 0.8984 |
| 30:70 | 1.0000 | 0.8610 | 0.8669 |

# Random Forest Mean Scores

| Train : Test Ratio | Mean Train | Mean CV | Test Score | Max Depth | N Estimators |
|---|---|---|---|---|---|
| 80:20 | 0.9995 | 0.9114 | 0.9247 | 32 | 150 |
| 50:50 | 0.9992 | 0.8915 | 0.8984 | 16 | 100 |
| 30:70 | 1.0000 | 0.8610 | 0.8669 | 16 | 100 |

# Decision Tree (SKLearn based on C4.5)

# Decision Trees 80/20

Dataset characteristics:

Number of examples in the dataset = 3190

Number of examples reserved for test set = 638
Number of examples reserved for training via 10 fold CV = 2552
Class Distribution Ratio (N : EI : IE) = 2 : 1 : 1
N features : 60, all categorical (DNA base pairs in 60 base pair long sequence)

Grid Search parameter space for Random Forest = {'max_depth': range(3, 20)}
best estimator parameters found = {'max_depth': 6}
best estimator mean training score = 0.9573321284086651
best estimator mean validation score = 0.9455329153605015
best estimator test score = 0.9404388714733543

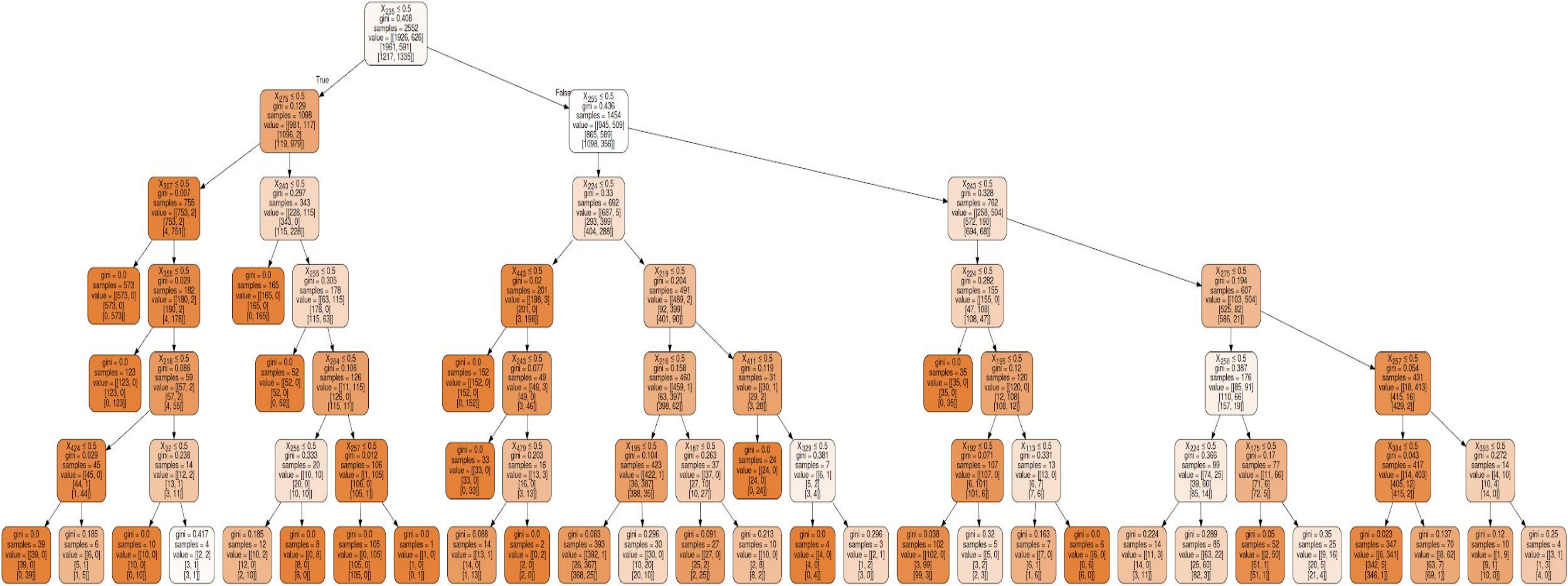
# Decision Tree Scores

| Train : Test Ratio | Mean Train | Mean CV | Test Score |
|---|---|---|---|
| 80:20 | 0.9570 | 0.9447 | 0.9341 |
| 50:50 | 0.9484 | 0.9347 | 0.9373 |
| 30:70 | 0.9564 | 0.9362 | 0.9305 |

# Decision Tree Best Parameters

| Train : Test Ratio | Max Depth (3-20) | Min samples split [3,5,25] |
|---|---|---|
| 80:20 | 5 | 25 |
| 50:50 | 5 | 3 |
| 30:70 | 6 | 3 |

# Resulting Decision Tree


$X_{235} \leq 0.5$ gini = 0.408 samples = 2552 value = [[1926, 626] [1961, 591] [1217, 1335]]
True
False
$X_{275} \leq 0.5$ gini = 0.129 samples = 1098 value = [[981, 117] [1096, 2] [119, 979]]
$X_{255} \leq 0.5$ gini = 0.436 samples = 1454 value = [[945, 509] [865, 589] [1098, 356]]
$X_{307} \leq 0.5$ gini = 0.007 samples = 755 value = [[753, 2] [753, 2] [4, 751]]
$X_{243} \leq 0.5$ gini = 0.297 samples = 343 value = [[228, 115] [343, 0] [115, 228]]
$X_{224} \leq 0.5$ gini = 0.33 samples = 692 value = [[687, 5] [293, 399] [404, 288]]
$X_{243} \leq 0.5$ gini = 0.328 samples = 762 value = [[258, 504] [572, 190] [694, 68]]
gini = 0.0 samples = 573 value = [[573, 0] [573, 0] [0, 573]]
$X_{355} \leq 0.5$ gini = 0.029 samples = 182 value = [[180, 2] [180, 2] [4, 178]]
gini = 0.0 samples = 165 value = [[165, 0] [165, 0] [0, 165]]
$X_{255} \leq 0.5$ gini = 0.305 samples = 178 value = [[63, 115] [178, 0] [115, 63]]
$X_{443} \leq 0.5$ gini = 0.02 samples = 201 value = [[198, 3] [201, 0] [3, 198]]
$X_{219} \leq 0.5$ gini = 0.204 samples = 491 value = [[489, 2] [92, 399] [401, 90]]
$X_{224} \leq 0.5$ gini = 0.282 samples = 155 value = [[155, 0] [47, 108] [108, 47]]
$X_{275} \leq 0.5$ gini = 0.194 samples = 607 value = [[103, 504] [525, 82] [586, 21]]
gini = 0.0 samples = 123 value = [[123, 0] [123, 0] [0, 123]]
$X_{216} \leq 0.5$ gini = 0.086 samples = 59 value = [[57, 2] [57, 2] [4, 55]]
gini = 0.0 samples = 52 value = [[52, 0] [52, 0] [0, 52]]
$X_{264} \leq 0.5$ gini = 0.106 samples = 126 value = [[11, 115] [126, 0] [115, 11]]
gini = 0.0 samples = 152 value = [[152, 0] [152, 0] [0, 152]]
$X_{243} \leq 0.5$ gini = 0.077 samples = 49 value = [[46, 3] [49, 0] [3, 46]]
$X_{216} \leq 0.5$ gini = 0.158 samples = 460 value = [[459, 1] [63, 397] [398, 62]]
$X_{411} \leq 0.5$ gini = 0.119 samples = 31 value = [[30, 1] [29, 2] [3, 28]]
gini = 0.0 samples = 35 value = [[35, 0] [35, 0] [0, 35]]
$X_{195} \leq 0.5$ gini = 0.12 samples = 120 value = [[120, 0] [12, 108] [108, 12]]
$X_{256} \leq 0.5$ gini = 0.387 samples = 176 value = [[85, 91] [110, 66] [157, 19]]
$X_{257} \leq 0.5$ gini = 0.054 samples = 431 value = [[18, 413] [415, 16] [429, 2]]
$X_{424} \leq 0.5$ gini = 0.029 samples = 45 value = [[45, 0] [44, 1] [1, 44]]
$X_{32} \leq 0.5$ gini = 0.238 samples = 14 value = [[12, 2] [13, 1] [3, 11]]
$X_{256} \leq 0.5$ gini = 0.333 samples = 20 value = [[10, 10] [20, 0] [10, 10]]
$X_{257} \leq 0.5$ gini = 0.012 samples = 106 value = [[1, 105] [106, 0] [105, 1]]
gini = 0.0 samples = 33 value = [[33, 0] [33, 0] [0, 33]]
$X_{479} \leq 0.5$ gini = 0.203 samples = 16 value = [[13, 3] [16, 0] [3, 13]]
$X_{195} \leq 0.5$ gini = 0.104 samples = 423 value = [[422, 1] [36, 387] [388, 35]]
$X_{167} \leq 0.5$ gini = 0.263 samples = 37 value = [[37, 0] [27, 10] [10, 27]]
gini = 0.0 samples = 24 value = [[24, 0] [24, 0] [0, 24]]
$X_{329} \leq 0.5$ gini = 0.381 samples = 7 value = [[6, 1] [5, 2] [3, 4]]
$X_{192} \leq 0.5$ gini = 0.071 samples = 107 value = [[107, 0] [6, 101] [101, 6]]
$X_{113} \leq 0.5$ gini = 0.331 samples = 13 value = [[13, 0] [6, 7] [7, 6]]
$X_{224} \leq 0.5$ gini = 0.366 samples = 99 value = [[74, 25] [39, 60] [85, 14]]
$X_{175} \leq 0.5$ gini = 0.17 samples = 77 value = [[11, 66] [71, 6] [72, 5]]
$X_{304} \leq 0.5$ gini = 0.043 samples = 417 value = [[14, 403] [405, 12] [415, 2]]
$X_{393} \leq 0.5$ gini = 0.272 samples = 14 value = [[4, 10] [10, 4] [14, 0]]
gini = 0.0 samples = 39 value = [[39, 0] [39, 0] [0, 39]]
gini = 0.185 samples = 6 value = [[6, 0] [5, 1] [1, 5]]
gini = 0.0 samples = 10 value = [[10, 0] [10, 0] [0, 10]]
gini = 0.417 samples = 4 value = [[2, 2] [3, 1] [3, 1]]
gini = 0.185 samples = 12 value = [[10, 2] [12, 0] [2, 10]]
gini = 0.0 samples = 8 value = [[0, 8] [8, 0] [8, 0]]
gini = 0.0 samples = 105 value = [[0, 105] [105, 0] [105, 0]]
gini = 0.0 samples = 1 value = [[1, 0] [1, 0] [0, 1]]
gini = 0.088 samples = 14 value = [[13, 1] [14, 0] [1, 13]]
gini = 0.0 samples = 2 value = [[0, 2] [2, 0] [2, 0]]
gini = 0.083 samples = 393 value = [[392, 1] [26, 367] [368, 25]]
gini = 0.296 samples = 30 value = [[30, 0] [10, 20] [20, 10]]
gini = 0.091 samples = 27 value = [[27, 0] [25, 2] [2, 25]]
gini = 0.213 samples = 10 value = [[10, 0] [2, 8] [8, 2]]
gini = 0.0 samples = 4 value = [[4, 0] [4, 0] [0, 4]]
gini = 0.296 samples = 3 value = [[2, 1] [1, 2] [3, 0]]
gini = 0.038 samples = 102 value = [[102, 0] [3, 99] [99, 3]]
gini = 0.32 samples = 5 value = [[5, 0] [3, 2] [2, 3]]
gini = 0.163 samples = 7 value = [[7, 0] [6, 1] [1, 6]]
gini = 0.0 samples = 6 value = [[6, 0] [0, 6] [6, 0]]
gini = 0.224 samples = 14 value = [[11, 3] [14, 0] [3, 11]]
gini = 0.289 samples = 85 value = [[63, 22] [25, 60] [82, 3]]
gini = 0.05 samples = 52 value = [[2, 50] [51, 1] [51, 1]]
gini = 0.35 samples = 25 value = [[9, 16] [20, 5] [21, 4]]
gini = 0.023 samples = 347 value = [[6, 341] [342, 5] [346, 1]]
gini = 0.137 samples = 70 value = [[8, 62] [63, 7] [69, 1]]
gini = 0.12 samples = 10 value = [[1, 9] [9, 1] [10, 0]]
gini = 0.25 samples = 4 value = [[1, 3] [1, 3] [4, 0]]

# SVM

# Support Vector Machines (20% Test)

```
Grid Search parameter space for SVM = [{'C': [100, 1000, 10000], 'gamma': [0.01, 0.001], 'kernel':
['poly', 'rbf'], 'degree': [2, 3, 4]}]
   best estimator parameters found = {'C': 100, 'degree': 3, 'gamma': 0.01, 'kernel': 'poly'}
   best estimator mean training score       = 0.9996081466330754
   best estimator mean validation score     = 0.9678683385579937
   best estimator test score                = 0.9702194357366771
   Process Time                             = 8.436292 Minutes
```

# Support Vector Machines (50% Test)

```
Grid Search parameter space for SVM = [{'C': [100, 1000, 10000], 'gamma': [0.01, 0.001], 'kernel':
['poly', 'rbf'], 'degree': [2, 3, 4]}]
   best estimator parameters found = {'C': 100, 'degree': 3, 'gamma': 0.01, 'kernel': 'poly'}
   best estimator mean training score       = 0.9994427513515088
   best estimator mean validation score     = 0.9605015673981191
   best estimator test score                = 0.9661442006269593
   Process Time                             = 3.845329 Minutes
```

# Support Vector Machines (70% Test)

```
Grid Search parameter space for SVM = [{'C': [100, 1000, 10000], 'gamma': [0.01, 0.001], 'kernel':
['poly', 'rbf'], 'degree': [2, 3, 4]}]
   best estimator parameters found = {'C': 100, 'degree': 3, 'gamma': 0.01, 'kernel': 'poly'}
   best estimator mean training score       = 1.0
   best estimator mean validation score     = 0.9540229885057471
   best estimator test score                = 0.961486789072996
   Process Time                             = 1.603256 Minutes
```

# Support Vector Machines Mean Scores

| Train : Test Ratio | Mean Train | Mean CV | Test Score |
|---|---|---|---|
| 80:20 | 0.9996 | 0.9679 | 0.9702 |
| 50:50 | 0.9994 | 0.9605 | 0.9661 |
| 30:70 | 1.0 | 0.9540 | 0.9615 |

# Neural Network

# Neural Network

- One layer, 50 nodes
- Scores 95.52±0.74% on stratified cross validation
- 20% test
  - 99.61% Train
  - 94.36% Test
- 50% test
  - 99.44% Train
  - 93.67 Test
- 70% test
  - 100% train
  - 91.49% test

# Performance Comparison

# Top Dog for 20% Test

| Algorithm | Best Test Score | Accompanying CV Score | Accompanying Train Score |
|---|---|---|---|
| Random Forest | 0.9247 | 0.9114 | 0.9995 |
| Decision Tree | 0.9373 | 0.9347 | 0.9484 |
| SVM 🐶🔥💯 | 0.9702 👑 | 0.9679 👑 | 0.9996 👑 |
| Neural Network | 0.9436 | 0.9552 | 0.9961 |

# Top Dog for 50% Test

| Algorithm | Best Test Score | Accompanying CV Score | Accompanying Train Score |
|---|---|---|---|
| Random Forest | 0.8984 | 0.8915 | 0.9992 |
| Decision Tree | 0.9373 | 0.9347 | 0.9484 |
| SVM 🐶🔥💯 | 0.9661 👑 | 0.9605 👑 | 0.9994 👑 |
| Neural Network | 0.9436 | 0.9552 | 0.9961 |

# Top Dog for 70% Test

| Algorithm | Best Test Score | Accompanying CV Score | Accompanying Train Score |
|---|---|---|---|
| Random Forest | 0.8669 | 0.8610 | 1.0000 👔 |
| Decision Tree | 0.9373 | 0.9347 | 0.9484 |
| SVM 🐶🔥💯 | 0.9615 👑 | 0.9540 | 1.0000 👔 |
| Neural Network | 0.9436 | 0.9552 👑 | 0.9961 |

# What could have been done differently?

- Investigated Gini importances of features
- Investigated other pattern recognition neural network architectures
- Investigated learning curves of models to better understand generalization behaviors
- Investigated Confusion matrices to see if class distribution was an issue that skewed different types of errors
- Used stratified K fold rather than random sample K fold
- Played more with parameters in general

# Thank you!

End of presentation