Compression Normal-Absolute Spambase Dataset

March 24, 2017

Compute the performance of MAB methods

```
In [1]: import numpy as np
        import time
        import sys
        import matplotlib.pyplot as plt
        from sklearn import metrics
        %matplotlib inline
        #plt.rcParams['figure.figsize'] = (15, 6)
```

0.1 Load BOKEH libariry

```
In [2]: from bokeh.layouts import row, gridplot
        from bokeh.plotting import figure, output_notebook, show
        from bokeh.models import Legend
        TOOLS = 'box_zoom,box_select,crosshair,resize,reset,lasso_select,pan,save,poly_select,ta
        output_notebook()
```

1 Compare the accuracy of the models

1.1 Load the pruned algorithm from normal prune

```
In [3]: ucb1 = np.load('/Users/salemameen/Desktop/banditsbook/python/UCB1/AccuracyAftrerPrune.np
        EpsilonGreedy = np.load('/Users/salemameen/Desktop/banditsbook/python/EpsilonGreedy/Accu
        AnnealingEpsilonGreedy = np.load('/Users/salemameen/Desktop/banditsbook/python/Annealing
        Softmax = np.load('/Users/salemameen/Desktop/banditsbook/python/Softmax/AccuracyAftrerPr
        AnnealingSoftmax = np.load('/Users/salemameen/Desktop/banditsbook/python/AnnealingSoftma
        Exp3 = np.load('/Users/salemameen/Desktop/banditsbook/python/Exp3/AccuracyAftrerPrune.np
        Hedge = np.load('/Users/salemameen/Desktop/banditsbook/python/Hedge/AccuracyAftrerPrune.
        ThompsonSampling = np.load('/Users/salemameen/Desktop/banditsbook/python/Thompson_Sampli
        Accuracy = np.load('/Users/salemameen/Desktop/banditsbook/python/AccuracyBeforePruning.n
```

1.2 Load the pruned algorithm from absulate prune

```
In [4]: ucb1_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output/python/UCB1
        EpsilonGreedy_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output/py
        AnnealingEpsilonGreedy_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_
```
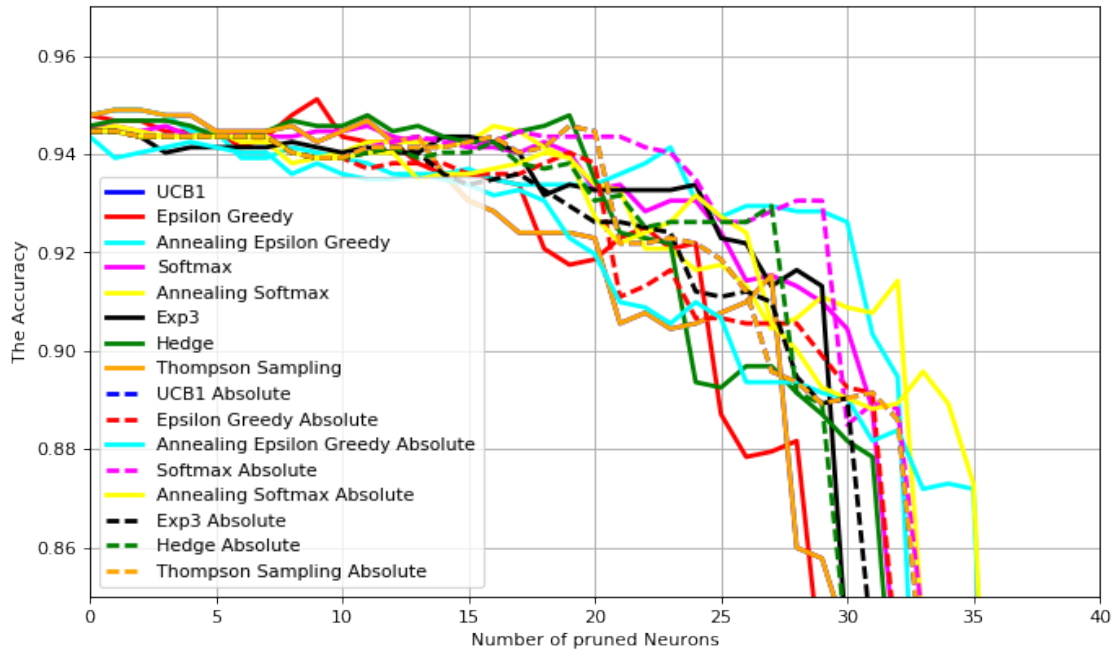
```python
Softmax_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output/python/S
AnnealingSoftmax_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output
Exp3_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output/python/Exp3
Hedge_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output/python/Hed
ThompsonSampling_absulate = np.load('/Users/salemameen/Desktop/banditsbook_moving_output
```

```python
In [5]: fig = plt.figure(figsize=(10, 6), dpi=80)
        ax = fig.add_subplot(111)
        N = len(ucb1)
        ## necessary variables
        ind = np.arange(N)                  # the x locations for the groups
        ### Normal algoritms
        plt.plot(ind , ucb1 , color="blue", linewidth=2.5, linestyle="-", label="UCB1")
        plt.plot(ind , EpsilonGreedy, color="red", linewidth=2.5, linestyle="-", label="Epsilon
        plt.plot(ind , AnnealingEpsilonGreedy, color="cyan", linewidth=2.5, linestyle="-", label
        plt.plot(ind , Softmax, color="magenta", linewidth=2.5, linestyle="-", label="Softmax")
        plt.plot(ind , AnnealingSoftmax, color="yellow", linewidth=2.5, linestyle="-", label="An
        plt.plot(ind , Exp3, color="black", linewidth=2.5, linestyle="-", label="Exp3")
        plt.plot(ind , Hedge, color="green", linewidth=2.5, linestyle="-", label="Hedge")
        plt.plot(ind , ThompsonSampling, color="orange", linewidth=2.5, linestyle="-", label="Th
        ### Absuluate once algoritms
        plt.plot(ind , ucb1_absulate , color="blue", linewidth=2.5, linestyle="--", label="UCB1
        plt.plot(ind , EpsilonGreedy_absulate, color="red", linewidth=2.5, linestyle="--", label
        plt.plot(ind , AnnealingEpsilonGreedy_absulate, color="cyan", linewidth=2.5, linestyle="
        plt.plot(ind , Softmax_absulate, color="magenta", linewidth=2.5, linestyle="--", label="
        plt.plot(ind , AnnealingSoftmax_absulate, color="yellow", linewidth=2.5, linestyle="-",
        plt.plot(ind , Exp3_absulate, color="black", linewidth=2.5, linestyle="--", label="Exp3
        plt.plot(ind , Hedge_absulate, color="green", linewidth=2.5, linestyle="--", label="Hedg
        plt.plot(ind , ThompsonSampling_absulate, color="orange", linewidth=2.5, linestyle="--",
        #########################
        plt.legend(loc = 3)
        plt.axis([0, 40, 0.85, 0.97])
        plt.xlabel('Number of pruned Neurons')
        plt.ylabel('The Accuracy')
        plt.grid(True)
        plt.show()
```

The plot legend contains:
- UCB1
- Epsilon Greedy
- Annealing Epsilon Greedy
- Softmax
- Annealing Softmax
- Exp3
- Hedge
- Thompson Sampling
- UCB1 Absolute
- Epsilon Greedy Absolute
- Annealing Epsilon Greedy Absolute
- Softmax Absolute
- Annealing Softmax Absolute
- Exp3 Absolute
- Hedge Absolute
- Thompson Sampling Absolute

Y-axis: The Accuracy
X-axis: Number of pruned Neurons

```python
In [6]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
        # Normal pruned
        #p1.circle(ind, ucb1, legend="ucb1", color="orange")
        p1.line(ind, ucb1, legend="ucb1", line_color="orange", line_width=2)
        #p1.square(ind, EpsilonGreedy, legend="Epsilon Greedy", fill_color=None, line_color="red
        p1.line(ind, EpsilonGreedy, legend="Epsilon Greedy", line_color="red", line_width=2)
        #p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="
        p1.line(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="blue
        #p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
        p1.line(ind, Softmax, legend="Softmax", line_color="green", line_width=2)
        #p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle=
        p1.line(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", line_width
        #p1.oval(ind, Exp3, legend="Exp3", line_color="black", height=0.01, width=0.01)
        p1.line(ind, Exp3, legend="Exp3", line_color="black", line_width=2)
        #p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
        #p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
        p1.line(ind, Hedge, legend="Hedge", line_color="yellow", line_width=2)
        #p1.square_cross(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink")
        p1.line(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink", line_width
        #p1.line(ind, Exp3, legend="2*sin(x)", line_dash=(4, 4), line_color="orange", line_width
        #p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
        ##### =========================================================================
        # Absulate pruned
        p1.circle(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="orange", line_width=2)
        p1.line(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="orange", line_width=2)
```

3

```
#p1.square(ind, EpsilonGreedy, legend="Epsilon Greedy", fill_color=None, line_color="red
p1.circle(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="red
p1.line(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="red",
#p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="
p1.circle(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolut
p1.line(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolute"
#p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
p1.circle(ind, Softmax_absulate, legend="Softmax Absolute", line_color="green", line_wid
p1.line(ind, Softmax_absulate, legend="Softmax Absolute", line_color="green", line_width
#p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle=
p1.circle(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_colo
p1.line(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_color=
#p1.oval(ind, Exp3, legend="Exp3", line_color="black", height=0.01, width=0.01)
p1.circle(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="black", line_width=2)
p1.line(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="black", line_width=2)
#p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
#p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
p1.circle(ind, Hedge_absulate, legend="Hedge Absolute", line_color="yellow", line_width=
p1.line(ind, Hedge_absulate, legend="Hedge Absolute", line_color="yellow", line_width=2)
#p1.square_cross(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink")
p1.circle(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_colo
p1.line(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_color=
#p1.line(ind, Exp3, legend="2*sin(x)", line_dash=(4, 4), line_color="orange", line_width
#p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
p1.title.align = "center"
show(p1)
#show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```

1.3   Comparing All algorithms with the model before pruning
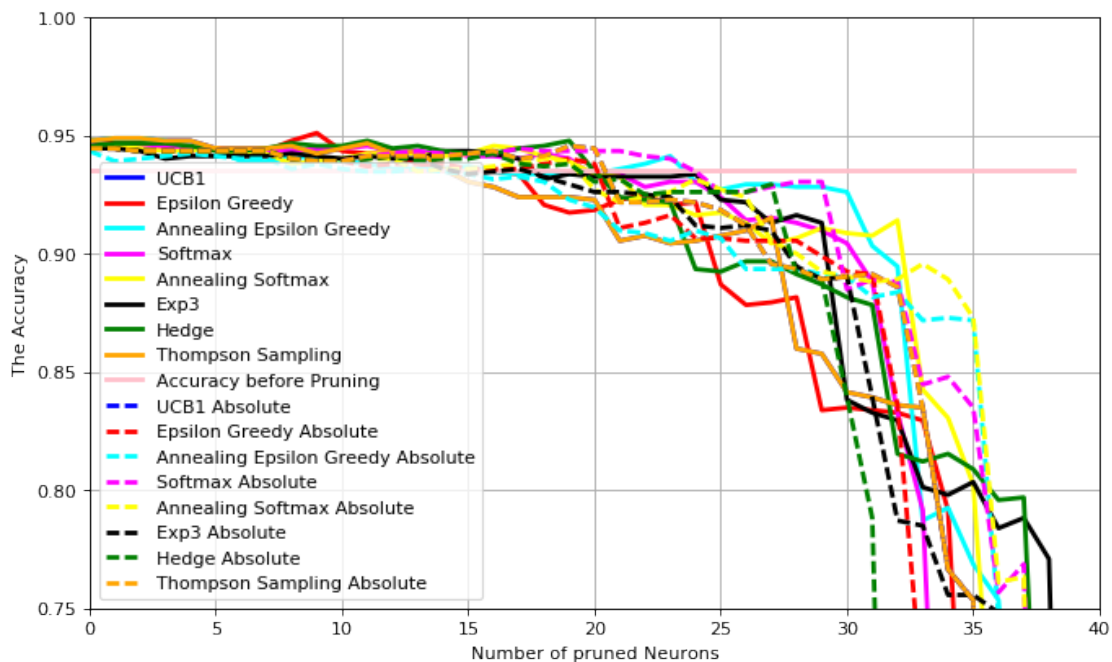
```
In [7]: fig = plt.figure(figsize=(10, 6), dpi=80)
        ax = fig.add_subplot(111)
        N = len(ucb1)
        Acc = [Accuracy for col in range(N)]
        ## necessary variables
        ind = np.arange(N)                     # the x locations for the groups
        plt.plot(ind , ucb1 , color="blue", linewidth=2.5, linestyle="-", label="UCB1")
        plt.plot(ind , EpsilonGreedy, color="red", linewidth=2.5, linestyle="-", label="Epsilon
        plt.plot(ind , AnnealingEpsilonGreedy, color="cyan", linewidth=2.5, linestyle="-", label
        plt.plot(ind , Softmax, color="magenta", linewidth=2.5, linestyle="-", label="Softmax")
        plt.plot(ind , AnnealingSoftmax, color="yellow", linewidth=2.5, linestyle="-", label="An
        plt.plot(ind , Exp3, color="black", linewidth=2.5, linestyle="-", label="Exp3")
        plt.plot(ind , Hedge, color="green", linewidth=2.5, linestyle="-", label="Hedge")
        plt.plot(ind , ThompsonSampling, color="orange", linewidth=2.5, linestyle="-", label="Th
        plt.plot(ind , Acc, color="pink", linewidth=2.5, linestyle="-", label="Accuracy before P
        ### Absuluate once algoritms
        plt.plot(ind , ucb1_absulate , color="blue", linewidth=2.5, linestyle="--", label="UCB1
        plt.plot(ind , EpsilonGreedy_absulate, color="red", linewidth=2.5, linestyle="--", label
```

4

```
plt.plot(ind , AnnealingEpsilonGreedy_absulate, color="cyan", linewidth=2.5, linestyle="
plt.plot(ind , Softmax_absulate, color="magenta", linewidth=2.5, linestyle="--", label="
plt.plot(ind , AnnealingSoftmax_absulate, color="yellow", linewidth=2.5, linestyle="--",
plt.plot(ind , Exp3_absulate, color="black", linewidth=2.5, linestyle="--", label="Exp3
plt.plot(ind , Hedge_absulate, color="green", linewidth=2.5, linestyle="--", label="Hedg
plt.plot(ind , ThompsonSampling_absulate, color="orange", linewidth=2.5, linestyle="--",
plt.legend(loc = 3)
plt.axis([0, 40, 0.75, 1])
plt.xlabel('Number of pruned Neurons')
plt.ylabel('The Accuracy')
plt.grid(True)
plt.show()
```



```
In [8]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
        # Normal pruned
        #p1.circle(ind, ucb1, legend="ucb1", color="orange")
        p1.line(ind, ucb1, legend="ucb1", line_color="orange", line_width=2)
        #p1.square(ind, EpsilonGreedy, legend="Epsilon Greedy", fill_color=None, line_color="red
        p1.line(ind, EpsilonGreedy, legend="Epsilon Greedy", line_color="red", line_width=2)
        #p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="
        p1.line(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="blue
        #p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
        p1.line(ind, Softmax, legend="Softmax", line_color="green", line_width=2)
        #p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle=
        p1.line(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", line_width
        #p1.oval(ind, Exp3, legend="Exp3", line_color="black", height=0.01, width=0.01)
```

```
p1.line(ind, Exp3, legend="Exp3", line_color="black", line_width=2)
#p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
#p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
p1.line(ind, Hedge, legend="Hedge", line_color="yellow", line_width=2)
#p1.square_cross(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink")
p1.line(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink", line_width
#p1.line(ind, Exp3, legend="2*sin(x)", line_dash=(4, 4), line_color="orange", line_width
#p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
##### ==================================================================
# Absulate pruned
# Absulate pruned
p1.circle(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="orange", line_width=2)
p1.line(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="orange", line_width=2)
#p1.square(ind, EpsilonGreedy, legend="Epsilon Greedy", fill_color=None, line_color="red
p1.circle(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="red"
p1.line(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="red",
#p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="
p1.circle(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolut
p1.line(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolute"
#p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
p1.circle(ind, Softmax_absulate, legend="Softmax Absolute", line_color="green", line_wid
p1.line(ind, Softmax_absulate, legend="Softmax Absolute", line_color="green", line_width
#p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle=
p1.circle(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_colo
p1.line(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_color=
#p1.oval(ind, Exp3, legend="Exp3", line_color="black", height=0.01, width=0.01)
p1.circle(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="black", line_width=2)
p1.line(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="black", line_width=2)
#p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
#p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
p1.circle(ind, Hedge_absulate, legend="Hedge Absolute", line_color="yellow", line_width=
p1.line(ind, Hedge_absulate, legend="Hedge Absolute", line_color="yellow", line_width=2)
#p1.square_cross(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink")
p1.circle(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_colo
p1.line(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_color=
p1.line(ind, Acc, legend="Accuracy", line_dash=(4, 4), line_color="orange", line_width=2
#p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
p1.title.align = "center"
show(p1)
#show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```

1.4 UCB1

```
In [9]: fig = plt.figure(figsize=(10, 6), dpi=80)
        ax = fig.add_subplot(111)
        N = len(ucb1)
        Acc = [Accuracy for col in range(N)]
        ## necessary variables
```

```python
ind = np.arange(N)                    # the x locations for the groups
plt.plot(ind , ucb1 , color="blue", linewidth=2.5, linestyle="-", label="UCB1")
plt.plot(ind , ucb1_absulate , color="green", linewidth=2.5, linestyle="-.", label="UCB1
plt.plot(ind , Acc, color="pink", linewidth=2.5, linestyle="-", label="Accuracy before P
plt.legend(loc = 3)
plt.axis([0, 40, 0.7, 1])
plt.xlabel('Number of pruned Neurons')
plt.ylabel('The Accuracy')
plt.grid(True)
plt.show()
```
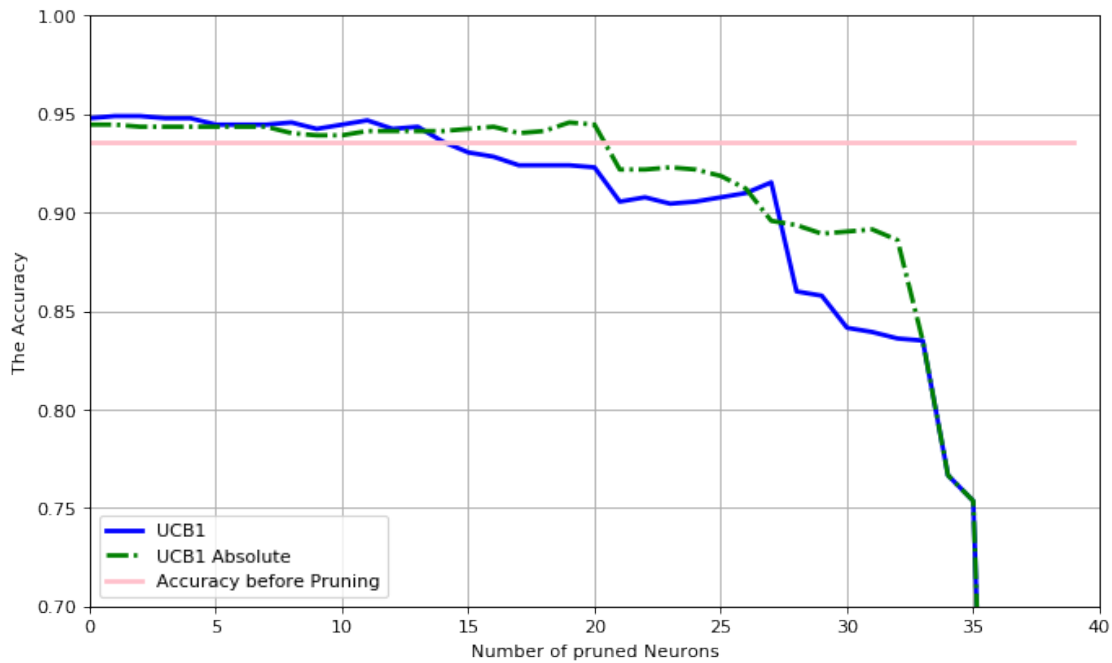


```python
In [10]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
         #p1.circle(ind, ucb1, legend="ucb1", color="orange")
         p1.line(ind, ucb1, legend="ucb1", line_color="blue", line_width=2)
         p1.circle(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="green", line_width=2)
         p1.line(ind, ucb1_absulate, legend="ucb1 Absolute", line_color="green", line_width=2)
         p1.line(ind, Acc, legend="Accuracy", line_dash=(4, 4), line_color="orange", line_width=
         #p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
         p1.title.align = "center"
         show(p1)
         #show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```

1.5   Epsilon greedy and Softmax
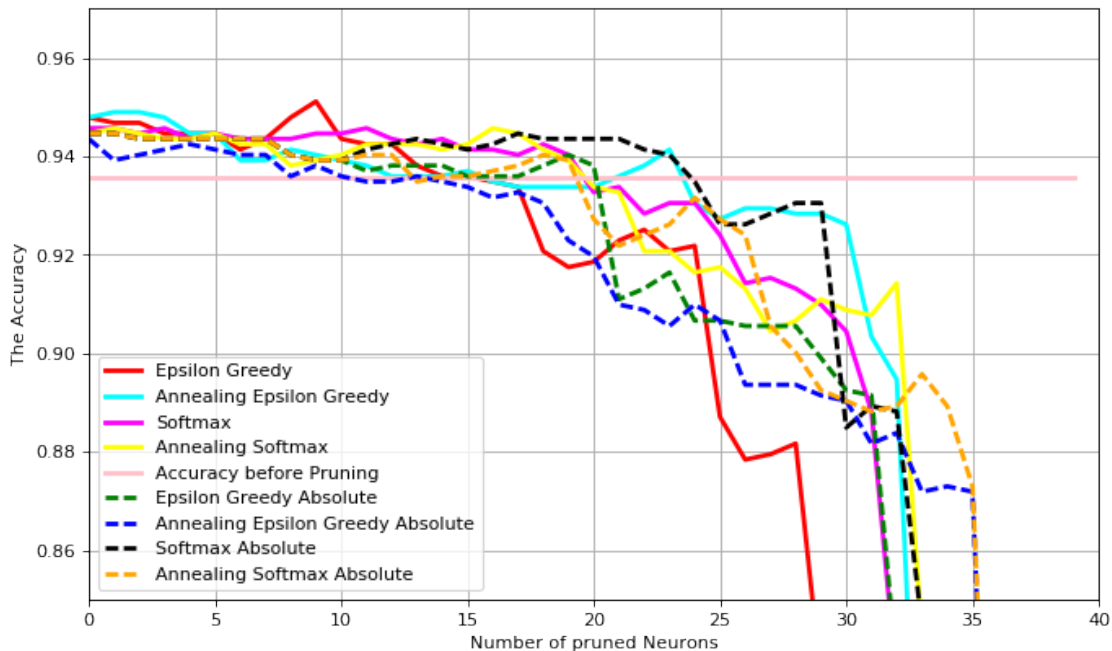
```python
In [11]: fig = plt.figure(figsize=(10, 6), dpi=80)
         ax = fig.add_subplot(111)
```

```
N = len(EpsilonGreedy)
Acc = [Accuracy for col in range(N)]
## necessary variables
ind = np.arange(N)                    # the x locations for the groups
plt.plot(ind , EpsilonGreedy, color="red", linewidth=2.5, linestyle="-", label="Epsilon
plt.plot(ind , AnnealingEpsilonGreedy, color="cyan", linewidth=2.5, linestyle="-", labe
plt.plot(ind , Softmax, color="magenta", linewidth=2.5, linestyle="-", label="Softmax")
plt.plot(ind , AnnealingSoftmax, color="yellow", linewidth=2.5, linestyle="-", label="A
plt.plot(ind , Acc, color="pink", linewidth=2.5, linestyle="-", label="Accuracy before
plt.plot(ind , EpsilonGreedy_absulate, color="green", linewidth=2.5, linestyle="--", la
plt.plot(ind , AnnealingEpsilonGreedy_absulate, color="blue", linewidth=2.5, linestyle=
plt.plot(ind , Softmax_absulate, color="black", linewidth=2.5, linestyle="--", label="S
plt.plot(ind , AnnealingSoftmax_absulate, color="orange", linewidth=2.5, linestyle="--"
plt.legend(loc = 3)
plt.axis([0, 40, 0.85, 0.97])
plt.xlabel('Number of pruned Neurons')
plt.ylabel('The Accuracy')
plt.grid(True)
plt.show()
```



```
In [12]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
         #p1.square(ind, EpsilonGreedy, legend="Epsilon Greedy", fill_color=None, line_color="re
         p1.line(ind, EpsilonGreedy, legend="Epsilon Greedy", line_color="red", line_width=2)
         #p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color=
         p1.line(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color="blu
         #p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
```

8

```
p1.line(ind, Softmax, legend="Softmax", line_color="green", line_width=2)
#p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle
p1.line(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", line_widt
p1.circle(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="or
p1.line(ind, EpsilonGreedy_absulate, legend="Epsilon Greedy Absolute", line_color="oran
#p1.ellipse(ind, AnnealingEpsilonGreedy, legend="Annealing Epsilon Greedy", line_color=
p1.circle(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolu
p1.line(ind, AnnealingEpsilonGreedy_absulate, legend="Annealing Epsilon Greedy Absolute
#p1.diamond(ind, Softmax, legend="Softmax", line_color="green")
p1.circle(ind, Softmax_absulate, legend="Softmax Absolute", line_color="cyan", line_wid
p1.line(ind, Softmax_absulate, legend="Softmax Absolute", line_color="cyan", line_width
#p1.arc(ind, AnnealingSoftmax, legend="Annealing Softmax", line_color="grey", end_angle
p1.circle(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_col
p1.line(ind, AnnealingSoftmax_absulate, legend="Annealing Softmax Absolute", line_color
p1.line(ind, Acc, legend="Accuracy", line_dash=(4, 4), line_color="black", line_width=2
#p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
p1.title.align = "center"
show(p1)
#show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```

1.6 Adversial Bandits Hedge and EXP3

```
In [13]: fig = plt.figure(figsize=(10, 6), dpi=80)
         ax = fig.add_subplot(111)
         N = len(Exp3)
         Acc = [Accuracy for col in range(N)]
         ## necessary variables
         ind = np.arange(N)                    # the x locations for the groups
         plt.plot(ind , Exp3, color="black", linewidth=2.5, linestyle="-", label="Exp3")
         plt.plot(ind , Hedge, color="green", linewidth=2.5, linestyle="-", label="Hedge")
         plt.plot(ind , Acc, color="pink", linewidth=2.5, linestyle="-", label="Accuracy before
         plt.plot(ind , Exp3_absulate, color="blue", linewidth=2.5, linestyle="--", label="Exp3
         plt.plot(ind , Hedge_absulate, color="red", linewidth=2.5, linestyle="--", label="Hedge
         plt.legend(loc = 3)
         plt.axis([0, 40, 0.9, 0.97])
         plt.xlabel('Number of pruned Neurons')
         plt.ylabel('The Accuracy')
         plt.grid(True)
         plt.show()
```
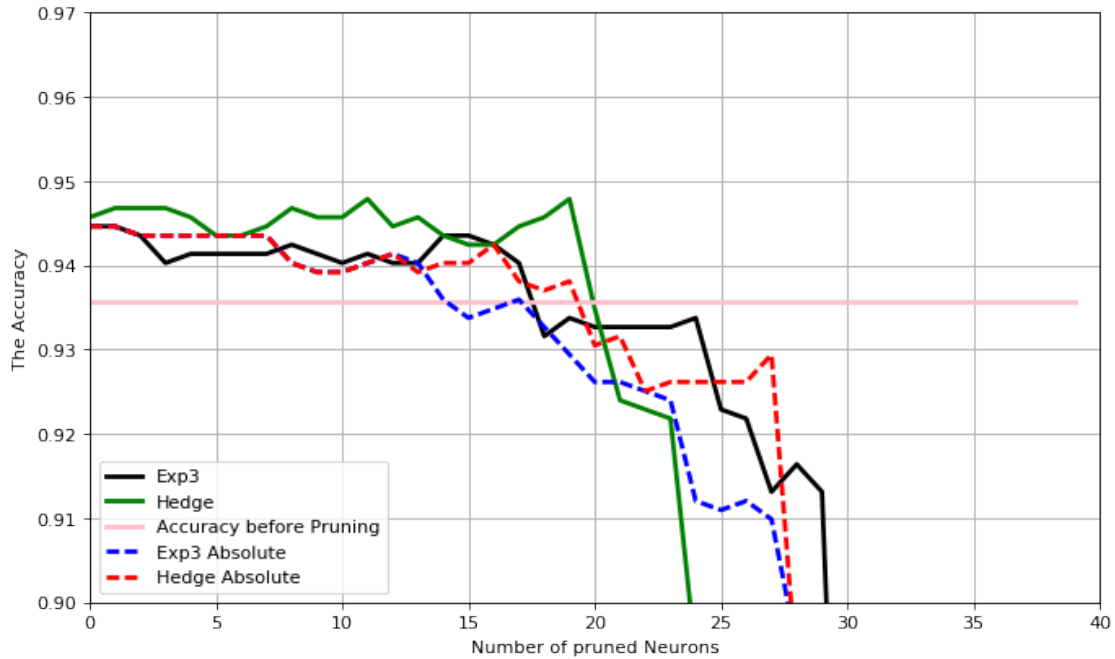
```
In [14]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
         #p1.oval(ind, Exp3, legend="Exp3", line_color="black", height=0.01, width=0.01)
         p1.line(ind, Exp3, legend="Exp3", line_color="black", line_width=2)
         #p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
         #p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
         p1.line(ind, Hedge, legend="Hedge", line_color="yellow", line_width=2)
         p1.circle(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="green", line_width=2)
         p1.line(ind, Exp3_absulate, legend="Exp3 Absolute", line_color="green", line_width=2)
         #p1.arc(ind, Hedge, legend="Hedge", line_color="yellow")
         #p1.triangle(ind, Hedge, legend="Hedge", line_color="yellow")
         p1.circle(ind, Hedge_absulate, legend="Hedge Absolute", line_color="red", line_width=2)
         p1.line(ind, Hedge_absulate, legend="Hedge Absolute", line_color="red", line_width=2)
         p1.line(ind, Acc, legend="Accuracy", line_dash=(4, 4), line_color="orange", line_width=
         #p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
         p1.title.align = "center"
         show(p1)
         #show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```
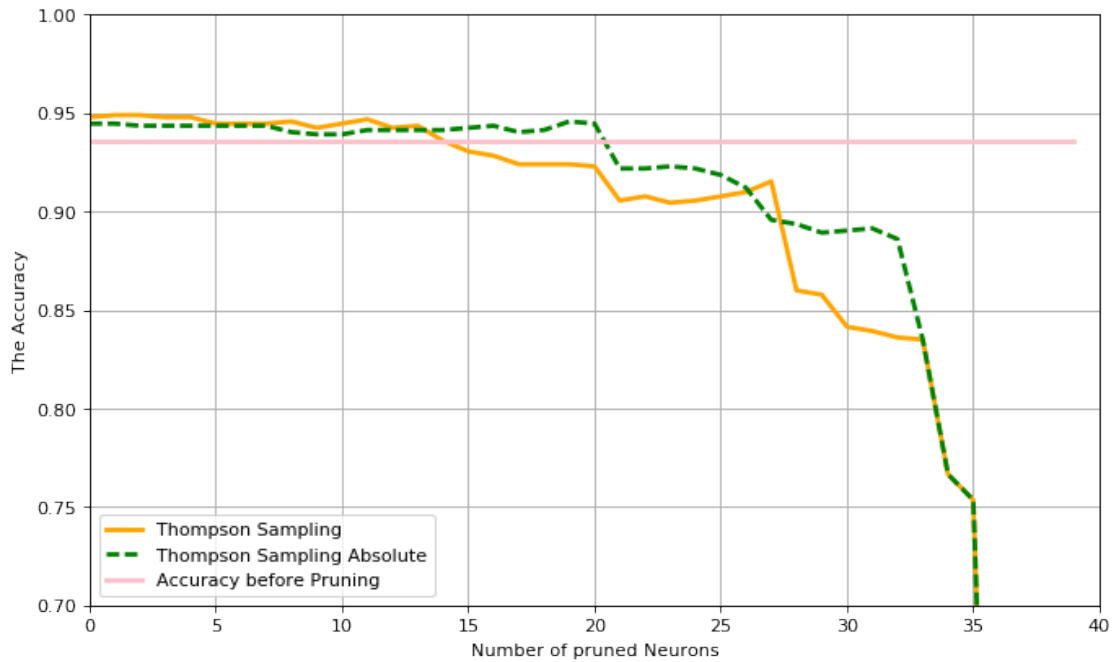
2 Thompson Sampling

```
In [15]: fig = plt.figure(figsize=(10, 6), dpi=80)
         ax = fig.add_subplot(111)
         N = len(ThompsonSampling)
         Acc = [Accuracy for col in range(N)]
         ## necessary variables
```

```
ind = np.arange(N)                          # the x locations for the groups
plt.plot(ind , ThompsonSampling, color="orange", linewidth=2.5, linestyle="-", label="T
plt.plot(ind , ThompsonSampling_absulate, color="green", linewidth=2.5, linestyle="--",
plt.plot(ind , Acc, color="pink", linewidth=2.5, linestyle="-", label="Accuracy before
plt.legend(loc = 3)
plt.axis([0, 40, 0.7, 1])
plt.xlabel('Number of pruned Neurons')
plt.ylabel('The Accuracy')
plt.grid(True)
plt.show()
```



```
In [16]: p1 = figure(title="The Performance over the number of neurons' pruned", tools=TOOLS)
         #p1.square_cross(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink")
         p1.line(ind, ThompsonSampling, legend="Thompson Sampling", line_color="pink", line_widt
         p1.line(ind, Acc, legend="Accuracy", line_dash=(4, 4), line_color="orange", line_width=
         p1.circle(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_col
         p1.line(ind, ThompsonSampling_absulate, legend="Thompson Sampling Absolute", line_color
         #p1.square(ind, Hedge, legend="3*sin(x)", fill_color=None, line_color="brown")
         p1.title.align = "center"
         show(p1)
         #show(gridplot(p1, p2, ncols=2, plot_width=400, plot_height=400))  # open a browser
```