Experiments for accuracy varying various hyper-parameters

Defining various Experiments.

List of Experiments hold each value as a list of # of Experiments, # of hidden units, learning rate

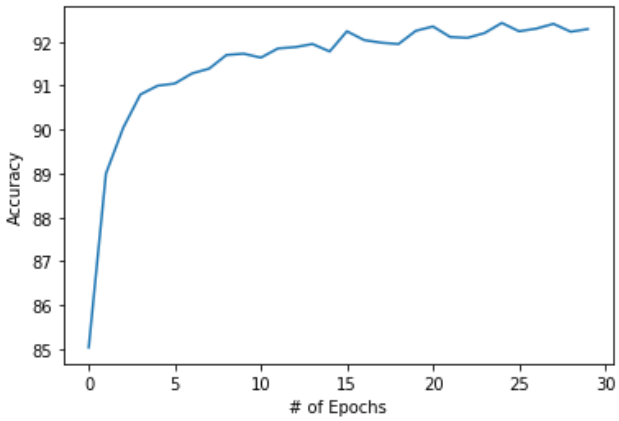
In [ ]:
```
from Network import callable
import numpy as np
import matplotlib.pyplot as plt

exp_all = []
exp_all.append([30,10,1.0])
exp_all.append([30,20,1.0])
exp_all.append([30,30,1.0])
exp_all.append([30,10,2.0])
exp_all.append([30,10,3.0])
exp_all.append([30,20,2.0])
exp_all.append([30,20,3.0])
exp_all.append([30,30,2.0])
exp_all.append([30,30,3.0])
```

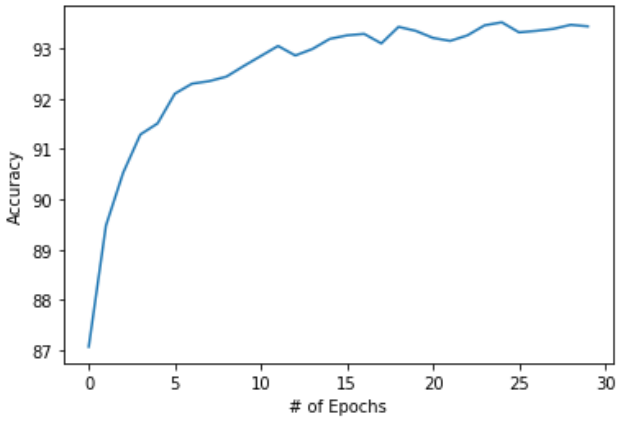
Run code for various value combination of hyper-parameter and plot graph between number of epochs and accuracy

In [2]:
```
all_exp_accuracy = []
for exp in exp_all:
    n_epochs = exp[0]
    hidden_units = exp[1]
    eta = exp[2]
    final_Accuracy = callable(n_epochs, hidden_units, eta)
    all_exp_accuracy.append(final_Accuracy)
```

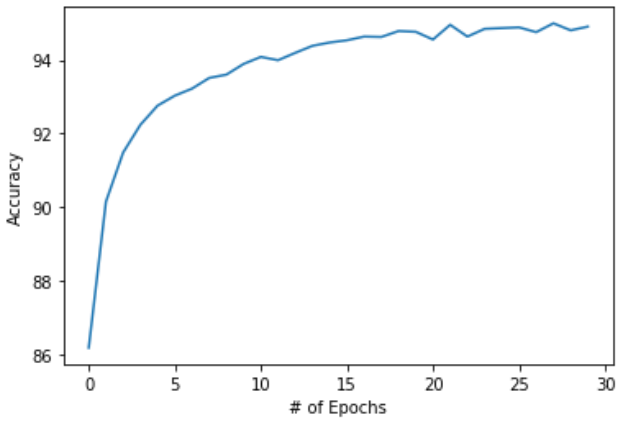
```
Epoch 29: 9229 / 10000
```



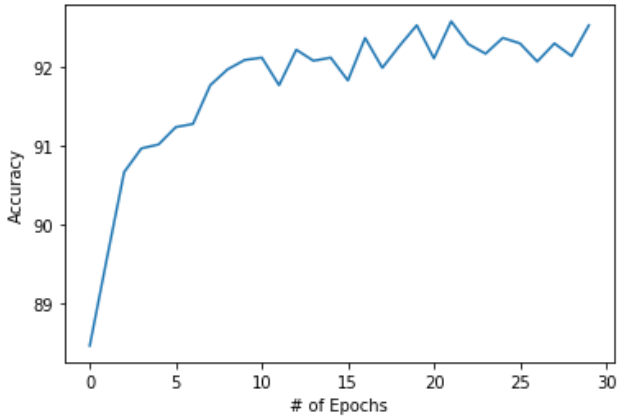

```
Epoch 29: 9344 / 10000
```



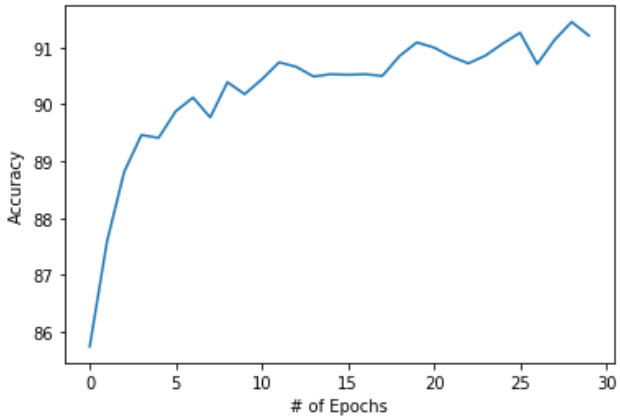

```
Epoch 29: 9490 / 10000
```



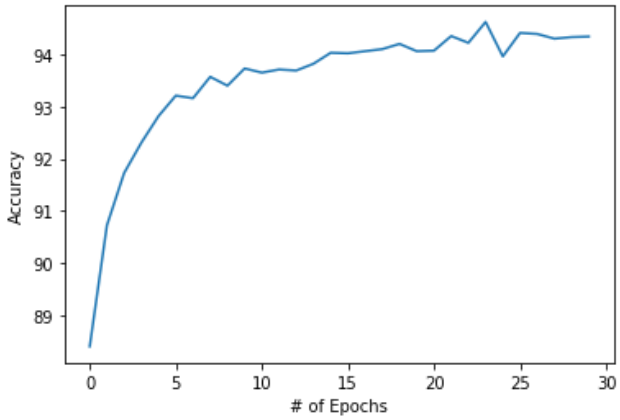

```
Epoch 29: 9253 / 10000
```



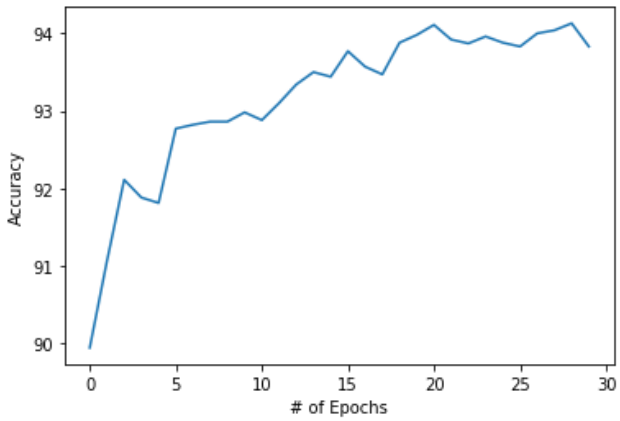

```
Epoch 29: 9121 / 10000
```



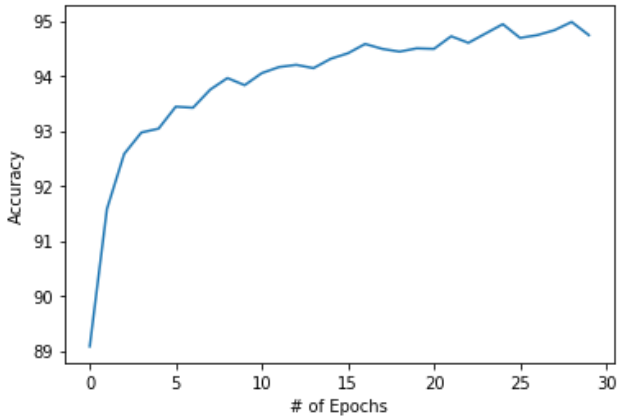

```
Epoch 29: 9434 / 10000
```



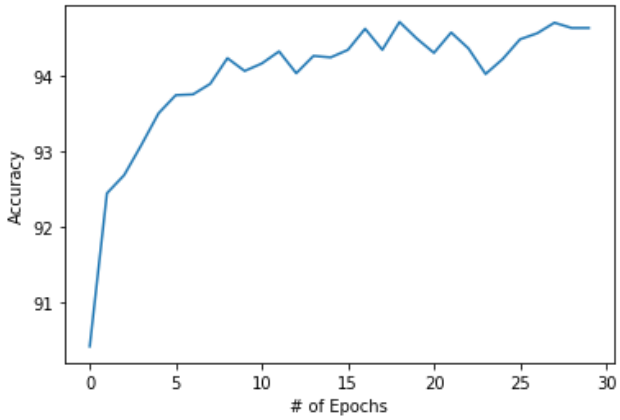

```
Epoch 29: 9383 / 10000
```




```
Epoch 29: 9474 / 10000
```

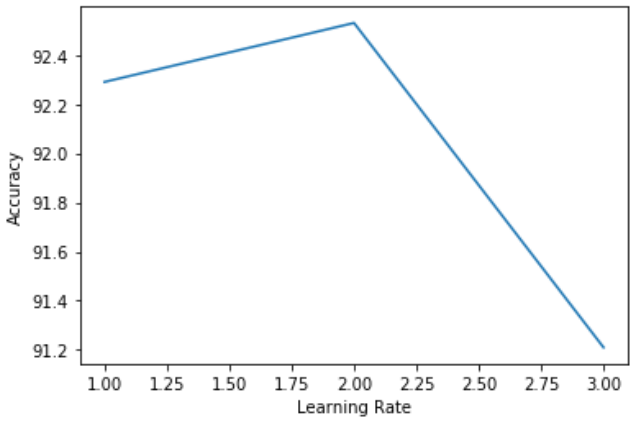



```
Epoch 29: 9464 / 10000
```



Collect all accuracys at the end of each experiment for the last epoch and develop a graph between learning rate and accuracy

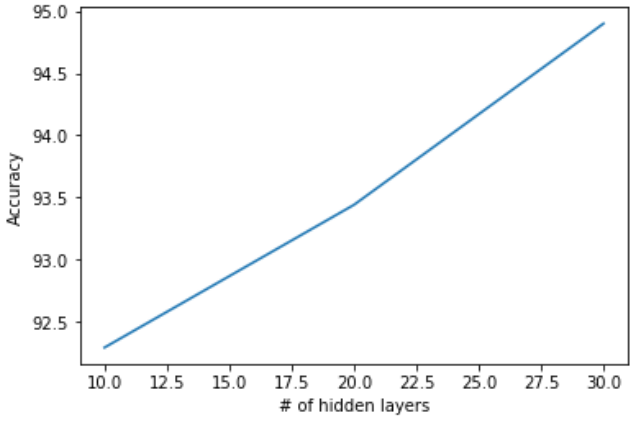
In [13]:
```
alpha = [1.,2.,3.]
alpha_accuracy = [all_exp_accuracy[0],all_exp_accuracy[3],all_exp_accuracy
[4]]
plt.plot(alpha,alpha_accuracy)
plt.xlabel('Learning Rate')
plt.ylabel('Accuracy')
plt.show()
```



Here we notice, as the learning rate is increased, accuracy increases. However, with further rise in learning rate, accuracy drops a lot. This is because, with higher value of learning rate, system misses the global minima and further starts rising in the graph.

Collect all accuracys at the end of each experiment for the last epoch and develop a graph between number of neurons in hidden layer and accuracy

In [6]:
```
n_hidden_units = [10,20,30]
neuron_accuracy = [all_exp_accuracy[0],all_exp_accuracy[1],all_exp_accuracy
[2]]
plt.plot(n_hidden_units,neuron_accuracy)
plt.xlabel('# of hidden layers')
plt.ylabel('Accuracy')
plt.show()
```



Here we notice, as the number of neurons in hidden layer increases, accuracy increases. This is because the system is growing in depth and multiplies the feauture computation.

With following configurations:

Configuration: 30-epochs, 30 hidden-units, 3.0 eta

On running the model 3-times, we get following values of accuracy

1st run: Epoch 29: 9477 / 10000

2nd run: Epoch 29: 9489 / 10000

3rd run: Epoch 29: 9504 / 10000

Thus for running our model 3 times with above configuration of 30 epochs, 30-hidden layer units, 3.0 eta, we get 94.89% Accuracy

As per the experiment, best accuracy is achieved always when number of hidden-layer units are increased

i.e. hidden-layer units = 30 number of epochs = 30 learning rate = 3.

We can further improve the prediction rate by increasing number of hidden layers so that computation can be increased on each pixel of the image and we get better accuracy.

In [ ]: