IE 345 - K “Introduction to Deep Learning: Fundamentals Concepts”

Prof. Yuzo

Classification

Random Forest

pg. 94 - 97

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:
```
dataset = pd.read_csv('C:/Users/pablo/Desktop/IE345_DeepLearning/DataAnalysisFromScratchw
ithPython_Peters Morgan/Datasets/Social_Network_Ads.csv')
x = dataset.iloc[:, [2, 3]].values
y = dataset.iloc[:, 4].values
dataset.head(10)
```

Out[2]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
5	15728773	Male	27	58000	0
6	15598044	Female	27	84000	0
7	15694829	Female	32	150000	1
8	15600575	Male	25	33000	0
9	15727311	Female	35	65000	0

In [3]:
```
# Splitting the dataset into the training and test set
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=0)
```

In [4]:
```
# Feature Scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
x_test = sc.transform(x_test)

# Fitting Random Forest Classification to the Training set
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators = 10, criterion = 'entropy', random_stat
e = 0)
classifier.fit(x_train, y_train)

# Predicting the Test set result
y_pred = classifier.predict(x_test)
```

```
C:\Users\pablo\Python\envs\DAVID\lib\site-packages\sklearn\utils\validation.py:595: Data
ConversionWarning: Data with input dtype int64 was converted to float64 by StandardScale
r.
  warnings.warn(msg, DataConversionWarning)
C:\Users\pablo\Python\envs\DAVID\lib\site-packages\sklearn\utils\validation.py:595: Data
ConversionWarning: Data with input dtype int64 was converted to float64 by StandardScale
r.
  warnings.warn(msg, DataConversionWarning)
C:\Users\pablo\Python\envs\DAVID\lib\site-packages\sklearn\utils\validation.py:595: Data
ConversionWarning: Data with input dtype int64 was converted to float64 by StandardScale
r.
  warnings.warn(msg, DataConversionWarning)
```

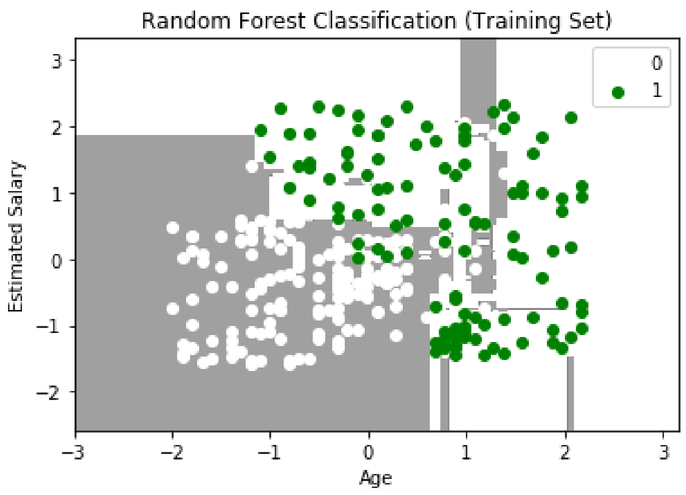
In [5]:
```
# Confusion Matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
```

```
In [6]: # Visualising the Training set result
        from matplotlib.colors import ListedColormap
        x_set, y_set = x_train, y_train
        X1, X2 = np.meshgrid(np.arange(start=x_set[:,0].min() - 1,
                                       stop=x_set[:,0].max() + 1,
                                       step = 0.01),
                             np.arange(start=x_set[:,1].min() - 1,
                                       stop=x_set[:,1].max() + 1,
                                       step = 0.01))

        plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.s
        hape),
                     alpha = 0.75, cmap = ListedColormap(('gray', 'white')))
        plt.xlim(X1.min(), X1.max())
        plt.ylim(X2.min(), X2.max())

        for i, j in enumerate(np.unique(y_set)):
            plt.scatter(x_set[y_set == j,0], x_set[y_set == j,1],
                        c = ListedColormap(('white', 'green'))(i), label = j)
        plt.title('Random Forest Classification (Training Set)')
        plt.xlabel('Age')
        plt.ylabel('Estimated Salary')
        plt.legend()
        plt.show()
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided a
s value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please
use a 2-D array with a single row if you really want to specify the same RGB or RGBA val
ue for all points.
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided a
s value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please
use a 2-D array with a single row if you really want to specify the same RGB or RGBA val
ue for all points.
```

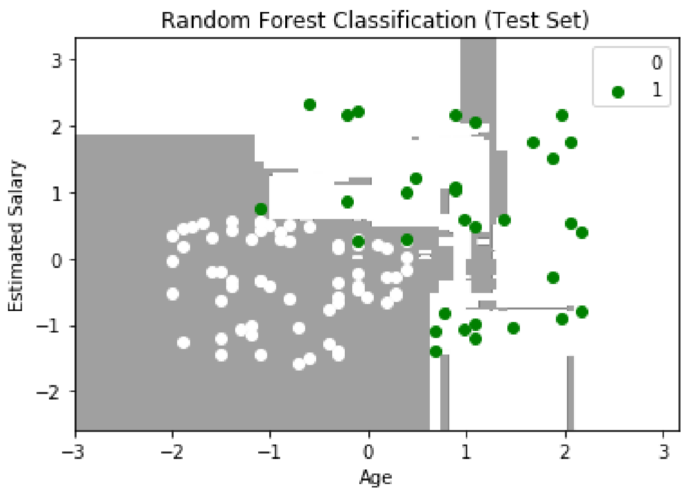


In [7]:
```python
# Visualising the Test set result
x_set, y_set = x_test, y_test
X1, X2 = np.meshgrid(np.arange(start=x_set[:,0].min() - 1,
                               stop=x_set[:,0].max() + 1,
                               step = 0.01),
                     np.arange(start=x_set[:,1].min() - 1,
                               stop=x_set[:,1].max() + 1,
                               step = 0.01))

plt.contourf(X1, X2, classifier.predict(np.array([X1.ravel(),X2.ravel()]).T).reshape(X1.s
hape),
            alpha = 0.75, cmap = ListedColormap(('gray', 'white')))
plt.xlim(X1.min(), X1.max())
plt.ylim(X2.min(), X2.max())

for i, j in enumerate(np.unique(y_set)):
    plt.scatter(x_set[y_set == j,0], x_set[y_set == j,1],
                c = ListedColormap(('white', 'green'))(i), label = j)
plt.title('Random Forest Classification (Test Set)')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')
plt.legend()
plt.show()
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided a
s value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please
use a 2-D array with a single row if you really want to specify the same RGB or RGBA val
ue for all points.
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided a
s value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please
use a 2-D array with a single row if you really want to specify the same RGB or RGBA val
ue for all points.
```



Pablo David Minango Negrete

pablodavid218@gmail.com

Lisber Arana Hinostroza

lisberarana@gmail.com