

The image shows a Jupyter Notebook interface with a dark theme. The top bar includes a 'Clear All Outputs' button, an 'Outline' tab, and a file browser showing 'mainnotebook.ipynb' with a Python kernel selected. The code cell contains the following Python code:

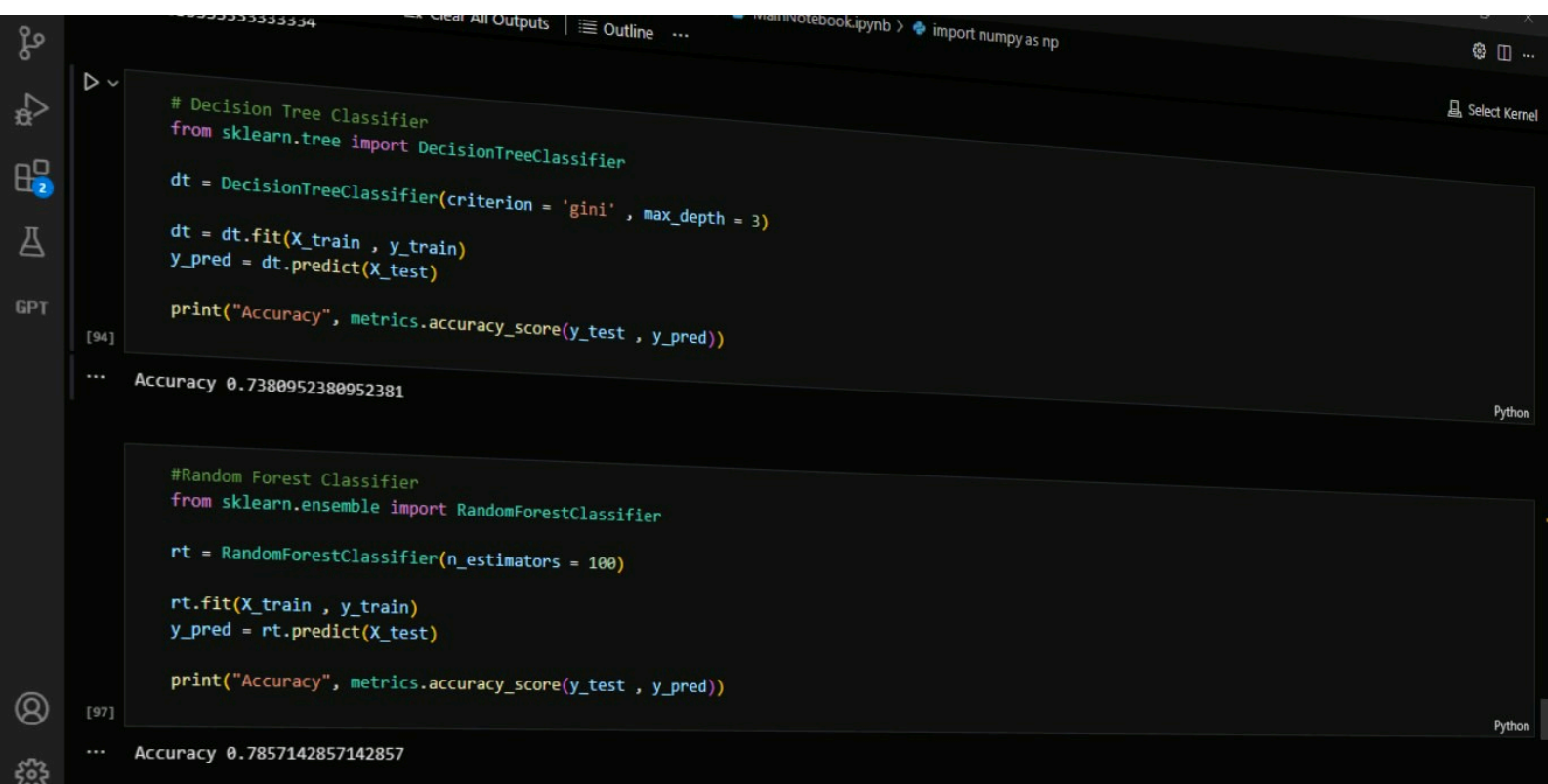
```
#Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier

rt = RandomForestClassifier(n_estimators = 100)

rt.fit(X_train , y_train)
y_pred = rt.predict(X_test)

print("Accuracy", metrics.accuracy_score(y_test , y_pred))
```

The cell is labeled '[97]' on the left. Below the code, the output is displayed as 'Accuracy 0.7857142857142857'. The interface also features a left sidebar with icons for file management, a 'GPT' button, and a bottom status bar with a user profile icon and a settings gear icon.



```
# Decision Tree Classifier
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier(criterion = 'gini' , max_depth = 3)

dt = dt.fit(X_train , y_train)
y_pred = dt.predict(X_test)

print("Accuracy", metrics.accuracy_score(y_test , y_pred))
```

[94]
... Accuracy 0.7380952380952381

```
#Random Forest Classifier
from sklearn.ensemble import RandomForestClassifier

rt = RandomForestClassifier(n_estimators = 100)

rt.fit(X_train , y_train)
y_pred = rt.predict(X_test)

print("Accuracy", metrics.accuracy_score(y_test , y_pred))
```

[97]
... Accuracy 0.7857142857142857

Clear All Outputs | Outline | mainnotebook.ipynb > import numpy as np

+ Code + Markdown

Select Kernel Python

Classification Models

```
 Logistic Regression -
 warnings
 warnings.filterwarnings('ignore')

 from sklearn.linear_model import LogisticRegression
 from sklearn import metrics

 logreg = LogisticRegression()

 logreg.fit(X_train , y_train)

 y_pred = logreg.predict(X_test)

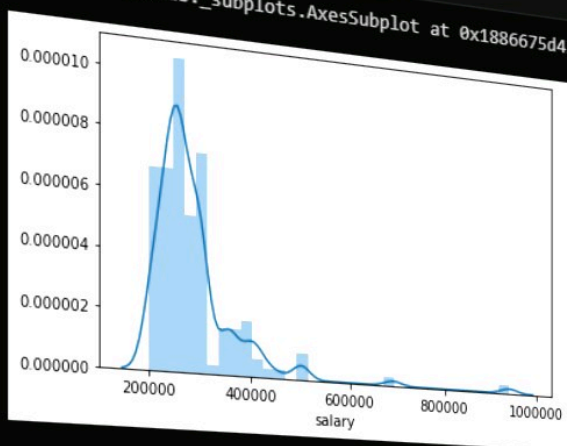
 print(logreg.score(X_test , y_test))

 .. 0.8333333333333334

 # Decision Tree Classifier
 from sklearn.tree import DecisionTreeClassifier

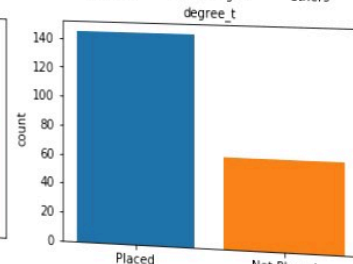
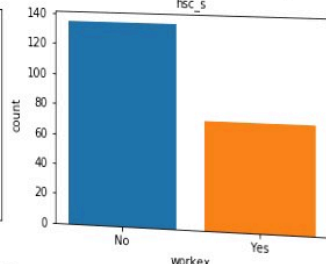
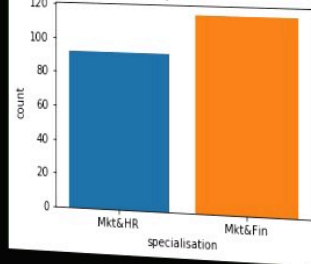
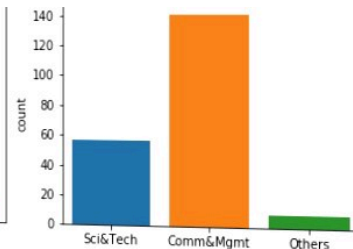
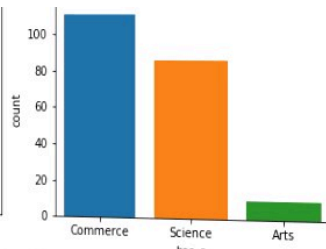
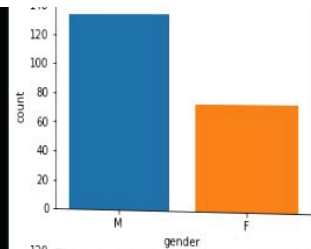
 dt = DecisionTreeClassifier(criterion = 'gini' , max_depth = 3)
```

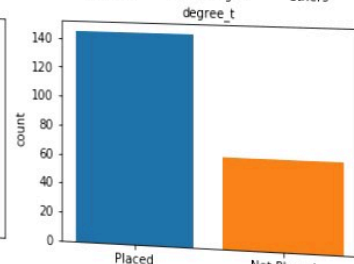
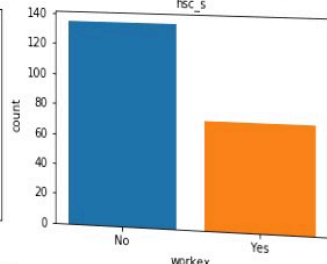
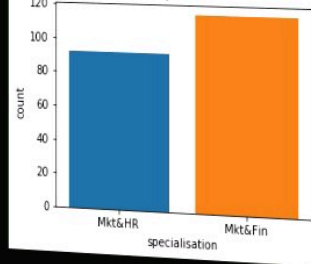
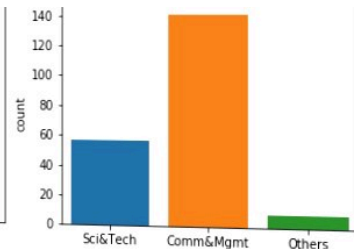
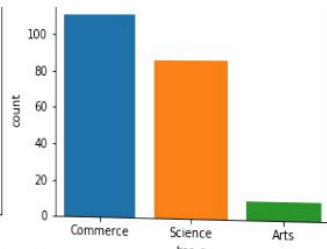
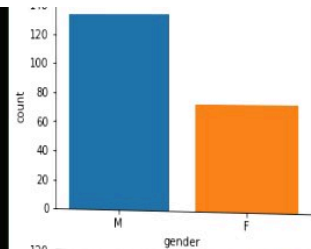
```
placement_placed['salary'])
placement_filtered.salary!= 0]
<matplotlib.axes._subplots.AxesSubplot at 0x1886675d438>
```



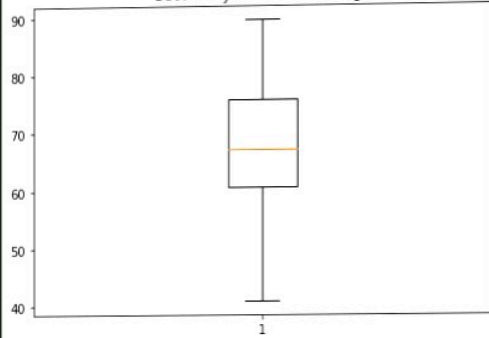
```
px.violin(placement_placed,y="salary",x="specialisation",color="gender",box=True,points="all")
```

Encoding

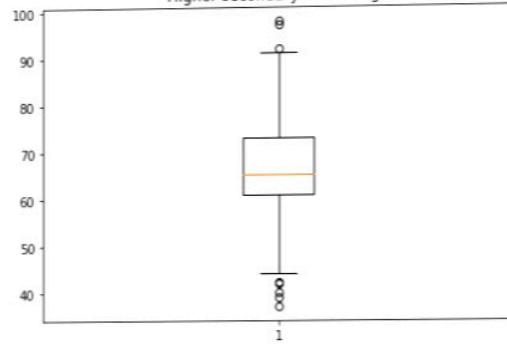




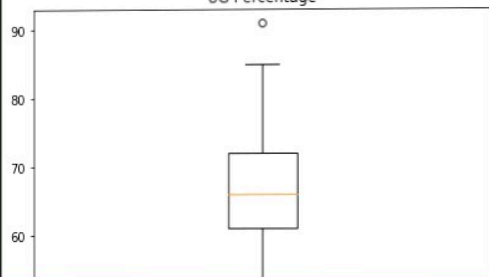
Secondary School Percentage



Higher secondary Percentage



UG Percentage



Employability Percentage

