Week 6 Assignment | Casey Masamitsu

In [456…

```
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
plt.rcParams['figure.figsize'] = (20, 6)
plt.rcParams['font.size'] = 14
import pandas as pd
```

In [457…

```
df = pd.read_csv('../data/adult.data', index_col=False)
```

In [458…

```
golden = pd.read_csv('../data/adult.test', index_col=False)
```

In [459…

```
df.head()
```

Out[459]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex
0	39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male
1	50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male
2	38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male
3	53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male
4	28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female

In [460…

```
golden.head()
```

Out[460]:

	age	workclass	fnlwgt	education	education-num	marital-status	occupation	relationship	race	sex
0	25	Private	226802	11th	7	Never-married	Machine-op-inspct	Own-child	Black	Male
1	38	Private	89814	HS-grad	9	Married-civ-spouse	Farming-fishing	Husband	White	Male
2	28	Local-gov	336951	Assoc-acdm	12	Married-civ-spouse	Protective-serv	Husband	White	Male
3	44	Private	160323	Some-college	10	Married-civ-spouse	Machine-op-inspct	Husband	Black	Male
4	18	?	103497	Some-college	10	Never-married	?	Own-child	White	Female

In [461…

```
df.columns
```

Out[461]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',
       'marital-status', 'occupation', 'relationship', 'race', 'sex',
       'capital-gain', 'capital-loss', 'hours-per-week', 'native-country',
       'salary'],
      dtype='object')
```

In [462…

```
from sklearn import preprocessing
```

In [463…

```
enc = preprocessing.OrdinalEncoder()
```

In [464…

```
transform_columns = ['sex']
non_num_columns = ['workclass', 'education', 'marital-status',
                   'occupation', 'relationship', 'race', 'sex',
                   'native-country']
```

In [465…

```
pd.get_dummies(df[transform_columns]).head()
```

Out[465]:

	sex_ Female	sex_ Male
0	0	1
1	0	1
2	0	1
3	0	1
4	1	0

In [466…

```
x = df.copy()

x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)

x["salary"] = enc.fit_transform(df[["salary"]])
```

In [467…

```
x.head()
```

Out[467]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_ Female	sex_ Male
0	39	77516	13	2174	0	40	0.0	0	1
1	50	83311	13	0	0	13	0.0	0	1
2	38	215646	9	0	0	40	0.0	0	1
3	53	234721	7	0	0	40	0.0	0	1
4	28	338409	13	0	0	40	0.0	1	0

In [468…

```
xt = golden.copy()

xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)

xt["salary"] = enc.fit_transform(golden[["salary"]])
```

In [469…

```
xt.head()
```

Out[469]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_ Female	sex_ Male
0	25	226802	7	0	0	40	0.0	0	1
1	38	89814	9	0	0	50	0.0	0	1
2	28	336951	12	0	0	40	1.0	0	1
3	44	160323	10	7688	0	40	1.0	0	1
4	18	103497	10	0	0	30	0.0	1	0

For the following use the above `adult` dataset. Start with only numerical features/columns.

In [470…

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
```

1. Show the RandomForest outperforms the DecisionTree for a fixed `max_depth` by training using the train set and `precision`, `recall`, `f1` on golden-test set.

Decision Tree Model

In [471…

```
dt_model = DecisionTreeClassifier(criterion='entropy', max_depth=5)
```

In [472…

```
dt_model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
```

Out[472]:

```
DecisionTreeClassifier(criterion='entropy', max_depth=5)
```

In [473…

```
dt_model.tree_.node_count
```

Out[473]:

```
49
```

In [474…

```
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, dt_model.feature_importances_))
```

Out[474]:

```
[('age', 0.2788779035969667),
 ('education-num', 0.2132324180244595),
 ('capital-gain', 0.3528567441438103),
 ('capital-loss', 0.042359393435638944),
 ('hours-per-week', 0.00842595230012558),
 ('sex_ Female', 0.03943299760989627),
 ('sex_ Male', 0.06481459088910276)]
```

In [475…

```
x.drop(['fnlwgt','salary'], axis=1).head()
```

Out[475]:

	age	education-num	capital-gain	capital-loss	hours-per-week	sex_ Female	sex_ Male
0	39	13	2174	0	40	0	1
1	50	13	0	0	13	0	1
2	38	9	0	0	40	0	1
3	53	7	0	0	40	0	1
4	28	13	0	0	40	1	0

In [476…

```
list(x.drop('salary', axis=1).columns)
```

Out[476]:
```
['age',
 'fnlwgt',
 'education-num',
 'capital-gain',
 'capital-loss',
 'hours-per-week',
 'sex_ Female',
 'sex_ Male']
```

In [477…
```
dt_predictions = dt_model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

Random Forest Model

In [478…
```
rf_model = RandomForestClassifier(criterion='entropy')
```

In [479…
```
rf_model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
```

Out[479]:
```
RandomForestClassifier(criterion='entropy')
```

In [480…
```
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[480]:
```
[('age', 0.22808023903209068),
 ('education-num', 0.04896252232865023),
 ('capital-gain', 0.09124106151534751),
 ('capital-loss', 0.03000626273288268),
 ('hours-per-week', 0.11540599503100893),
 ('sex_ Female', 0.007386355203465439),
 ('sex_ Male', 0.006465248304175578)]
```

In [481…
```
x.drop(['fnlwgt','salary'], axis=1).head()
```

Out[481]:

	age	education-num	capital-gain	capital-loss	hours-per-week	sex_ Female	sex_ Male
0	39	13	2174	0	40	0	1
1	50	13	0	0	13	0	1
2	38	9	0	0	40	0	1
3	53	7	0	0	40	0	1
4	28	13	0	0	40	1	0

In [482…
```
rf_predictions = rf_model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

Decision Tree (dt_model) vs Random Forest (rf_model) Performance

In [483…
```
from sklearn.metrics import (
    accuracy_score,
```

```
    classification_report,
    confusion_matrix, auc, roc_curve
)
```

Accuracy Scores

In [484…

```
accuracy_score(xt.salary, dt_predictions)
```

Out[484]:

```
0.8201584669246361
```

In [485…

```
accuracy_score(xt.salary, rf_predictions)
```

Out[485]:

```
0.8282046557336773
```

Confusion Matrices

In [486…

```
confusion_matrix(xt.salary, dt_predictions)
```

Out[486]:

```
array([[11458,   977],
       [ 1951,  1895]], dtype=int64)
```

In [487…

```
confusion_matrix(xt.salary, rf_predictions)
```

Out[487]:

```
array([[11483,   952],
       [ 1845,  2001]], dtype=int64)
```

Classifcation Reports

In [488…

```
print(classification_report(xt.salary, dt_predictions))
```

```
              precision    recall  f1-score   support

         0.0       0.85      0.92      0.89     12435
         1.0       0.66      0.49      0.56      3846

    accuracy                           0.82     16281
   macro avg       0.76      0.71      0.73     16281
weighted avg       0.81      0.82      0.81     16281
```

In [489…

```
print(classification_report(xt.salary, rf_predictions))
```

```
              precision    recall  f1-score   support

         0.0       0.86      0.92      0.89     12435
         1.0       0.68      0.52      0.59      3846

    accuracy                           0.83     16281
   macro avg       0.77      0.72      0.74     16281
weighted avg       0.82      0.83      0.82     16281
```

We can see that the random forest model has a slightly better accuracy score, precision, recall, and f1-score with max_depth set to "None." However, when playing around with max_depth, the decision tree model *sometimes* performs better in terms of precision, recall, and accuracy score (especially for predicting 0 values). I noticed this when setting the max_depth to 10 instead of "None."

2. For RandomForest or DecisionTree and using the `adult` dataset, systematically add new columns, one by one, that are non-numerical but converted using the feature-extraction techniques we learned. Show `[precision, recall, f1]` for each additional feature added.

In [490…

```
df.columns
```

Out[490]:

```
Index(['age', 'workclass', 'fnlwgt', 'education', 'education-num',
       'marital-status', 'occupation', 'relationship', 'race', 'sex',
       'capital-gain', 'capital-loss', 'hours-per-week', 'native-country',
       'salary'],
      dtype='object')
```

In [491…

```
enc = preprocessing.OrdinalEncoder()
```

Add Sex and Workclass

In [492…

```
transform_columns = ['sex', "workclass"]
non_num_columns = ["workclass", 'education', 'marital-status',
                   'occupation', 'relationship', 'race', 'sex',
                   'native-country']
# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
```

In [493…

```
x.head()
```

Out[493]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_ Female	sex_ Male	workclass_ ?	workclass_ Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex and Workclass

In [494…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[494]:

```
[('age', 0.3578346198753116),
 ('education-num', 0.16714863654436857),
 ('capital-gain', 0.15574397505123572),
 ('capital-loss', 0.058843102084920135),
 ('hours-per-week', 0.17207369484373486),
 ('sex_ Female', 0.025287910156278223),
 ('sex_ Male', 0.0223389874658935),
 ('workclass_ ?', 0.002784716669753499),
 ('workclass_ Federal-gov', 0.0033027821356796175),
 ('workclass_ Local-gov', 0.0035789583801502873),
 ('workclass_ Never-worked', 4.299550160921079e-06),
 ('workclass_ Private', 0.004957590185406713),
 ('workclass_ Self-emp-inc', 0.0064896020212075875),
 ('workclass_ Self-emp-not-inc', 0.004740190443136497),
 ('workclass_ State-gov', 0.0023502028678520186),
 ('workclass_ Without-pay', 5.5131885198411036e-05),
 ('workclass', 0.012465599839711813)]
```

In [495…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [496…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8224924758921442

Confusion Matrix
[[11364  1071]
 [ 1819  2027]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.86      0.91      0.89     12435
         1.0       0.65      0.53      0.58      3846

    accuracy                           0.82     16281
   macro avg       0.76      0.72      0.74     16281
weighted avg       0.81      0.82      0.82     16281
```

Add Education

In [497…

```
transform_columns = ['sex', "workclass", "education"]
non_num_columns = ["workclass", "education", 'marital-status',
                    'occupation', 'relationship', 'race', 'sex',
                    'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
```

In [498…

```
x.head()
```

Out[498]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_ Female	sex_ Male	workclass_ ?	workclass_ Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, and Education

In [499…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[499]:

```
[('age', 0.37712541088492213),
 ('education-num', 0.06630092188110635),
 ('capital-gain', 0.14947892381769964),
 ('capital-loss', 0.05573334364930979),
 ('hours-per-week', 0.18267996962308128),
 ('sex_ Female', 0.023681445196505838),
 ('sex_ Male', 0.02856132010540958),
 ('workclass_ ?', 0.003192012316178854),
 ('workclass_ Federal-gov', 0.003338537458316541),
 ('workclass_ Local-gov', 0.0035884468034333003),
 ('workclass_ Never-worked', 7.105008911736126e-06),
 ('workclass_ Private', 0.005396680457001168),
 ('workclass_ Self-emp-inc', 0.005489726817667985),
 ('workclass_ Self-emp-not-inc', 0.004988682473539063),
 ('workclass_ State-gov', 0.002676866535827987),
 ('workclass_ Without-pay', 6.27750237232366e-05),
 ('education_ 10th', 0.001213693277528152),
 ('education_ 11th', 0.0015217969175150723),
 ('education_ 12th', 0.0004850903845914436),
 ('education_ 1st-4th', 0.0002788814099195791),
 ('education_ 5th-6th', 0.000731738069644515),
 ('education_ 7th-8th', 0.0010949447473323941),
 ('education_ 9th', 0.0006881606102838893),
 ('education_ Assoc-acdm', 0.0012338070243023277),
 ('education_ Assoc-voc', 0.0017183649001333936),
 ('education_ Bachelors', 0.00914140890641336),
 ('education_ Doctorate', 0.004583055821881394),
 ('education_ HS-grad', 0.007660404665140993),
 ('education_ Masters', 0.006775555122948694),
 ('education_ Preschool', 0.000274640063418163),
 ('education_ Prof-school', 0.004621242951519769),
 ('education_ Some-college', 0.003645727620273618),
 ('workclass', 0.014150043028618475),
 ('education', 0.027879276425900223)]
```

In [500…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [501…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8193599901725939

Confusion Matrix
[[11336  1099]
 [ 1842  2004]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.86      0.91      0.89     12435
         1.0       0.65      0.52      0.58      3846

    accuracy                           0.82     16281
   macro avg       0.75      0.72      0.73     16281
weighted avg       0.81      0.82      0.81     16281
```

Add marital-status

In [502…

```
transform_columns = ['sex', "workclass", "education", "marital-status"]
non_num_columns = ["workclass", "education", "marital-status",
                   'occupation', 'relationship', 'race', 'sex', 'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])
x["marital-status"] = enc.fit_transform(df[["marital-status"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
xt["marital-status"] = enc.fit_transform(golden[["marital-status"]])
```

In [503…

```
x.head()
```

Out[503]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_Female	sex_Male	workclass_?	workclass_Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, Education, and Marital Status

In [504…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[504]:
```
[('age', 0.27924922758616),
 ('education-num', 0.05741613645658248),
 ('capital-gain', 0.12914417233831263),
 ('capital-loss', 0.04270995761549592 5),
 ('hours-per-week', 0.148060356238091),
 ('sex_ Female', 0.012844970064049112),
 ('sex_ Male', 0.014534792341857861),
 ('workclass_ ?', 0.0031031906493387508),
 ('workclass_ Federal-gov', 0.0038994949119447865),
 ('workclass_ Local-gov', 0.004181961379927724),
 ('workclass_ Never-worked', 1.4044450513947528e-05),
 ('workclass_ Private', 0.00632521036226319),
 ('workclass_ Self-emp-inc', 0.004938772719995758 5),
 ('workclass_ Self-emp-not-inc', 0.005682527520231948),
 ('workclass_ State-gov', 0.002855112257002532),
 ('workclass_ Without-pay', 8.989910650709871e-05),
 ('education_ 10th', 0.001021018264602996),
 ('education_ 11th', 0.0017064001052467766),
 ('education_ 12th', 0.0005410007743481649),
 ('education_ 1st-4th', 0.0001965057741442842),
 ('education_ 5th-6th', 0.0007143963814514527),
 ('education_ 7th-8th', 0.0014690082239064211),
 ('education_ 9th', 0.00080028031381266),
 ('education_ Assoc-acdm', 0.0013045422842861234),
 ('education_ Assoc-voc', 0.0016371142658600803),
 ('education_ Bachelors', 0.00985213948065431),
 ('education_ Doctorate', 0.0032575575959601696),
 ('education_ HS-grad', 0.008994864015247312),
 ('education_ Masters', 0.007306466713602739),
 ('education_ Preschool', 0.00029559877963348673),
 ('education_ Prof-school', 0.004338128759095464 6),
 ('education_ Some-college', 0.00305763359771387),
 ('marital-status_ Divorced', 0.01147681670447878),
 ('marital-status_ Married-AF-spouse', 0.0007003121117404787),
 ('marital-status_ Married-civ-spouse', 0.09629944371451454),
 ('marital-status_ Married-spouse-absent', 0.001137994544843239),
 ('marital-status_ Never-married', 0.03384942392988421),
 ('marital-status_ Separated', 0.002283022426190928),
 ('marital-status_ Widowed', 0.0017357131321021626),
 ('workclass', 0.01408524912313963),
 ('education', 0.026650902770422914),
 ('marital-status', 0.05023864021484212)]
```

In [505…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [506…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8364351084085744

Confusion Matrix
[[11306  1129]
 [ 1534  2312]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.88      0.91      0.89     12435
         1.0       0.67      0.60      0.63      3846

    accuracy                           0.84     16281
   macro avg       0.78      0.76      0.76     16281
weighted avg       0.83      0.84      0.83     16281
```

Add Occupation

In [507…

```
transform_columns = ['sex', "workclass", "education", "marital-status", "occupation"]
non_num_columns = ["workclass", "education", "marital-status", "occupation",
                   'relationship', 'race', 'sex', 'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])
x["marital-status"] = enc.fit_transform(df[["marital-status"]])
x["occupation"] = enc.fit_transform(df[["occupation"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
xt["marital-status"] = enc.fit_transform(golden[["marital-status"]])
xt["occupation"] = enc.fit_transform(golden[["occupation"]])
```

In [508…

```
x.head()
```

Out[508]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_Female	sex_Male	workclass_?	workclass_Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, Education, Marital Status, and Occupation

In [509…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[509]:
```
[('age', 0.26539328763899445),
 ('education-num', 0.055135645244753506),
 ('capital-gain', 0.1032851330769881),
 ('capital-loss', 0.03364621651518385),
 ('hours-per-week', 0.12975051263987394),
 ('sex_ Female', 0.012432896409999637),
 ('sex_ Male', 0.012688641829711059),
 ('workclass_ ?', 0.0011294555218839023),
 ('workclass_ Federal-gov', 0.0037177939833240377),
 ('workclass_ Local-gov', 0.00459133211207361),
 ('workclass_ Never-worked', 4.494753914542715e-06),
 ('workclass_ Private', 0.008013625667549876),
 ('workclass_ Self-emp-inc', 0.0050163100692006),
 ('workclass_ Self-emp-not-inc', 0.006034389173282301),
 ('workclass_ State-gov', 0.0032661710879253116),
 ('workclass_ Without-pay', 5.6375138065704846e-05),
 ('education_ 10th', 0.0009043450431522939),
 ('education_ 11th', 0.001445841316864122),
 ('education_ 12th', 0.0006459760554029936),
 ('education_ 1st-4th', 0.0002157175493187645),
 ('education_ 5th-6th', 0.0006310143099825745),
 ('education_ 7th-8th', 0.0013746623976809936),
 ('education_ 9th', 0.0007979427908440442),
 ('education_ Assoc-acdm', 0.001813213117192165),
 ('education_ Assoc-voc', 0.002177551414983966),
 ('education_ Bachelors', 0.008491461066341134),
 ('education_ Doctorate', 0.002830697316224548),
 ('education_ HS-grad', 0.006964111243485923),
 ('education_ Masters', 0.0061332589772861166),
 ('education_ Preschool', 0.00017270538078374294),
 ('education_ Prof-school', 0.0027944958573726344),
 ('education_ Some-college', 0.003958180716492268),
 ('marital-status_ Divorced', 0.010931148104519181),
 ('marital-status_ Married-AF-spouse', 0.0005580770950326694),
 ('marital-status_ Married-civ-spouse', 0.08209958391556431),
 ('marital-status_ Married-spouse-absent', 0.0010167904002539824),
 ('marital-status_ Never-married', 0.03426244828176309),
 ('marital-status_ Separated', 0.0020355038059009594),
 ('marital-status_ Widowed', 0.0017123914243675905),
 ('occupation_ ?', 0.0011454714285188752),
 ('occupation_ Adm-clerical', 0.003842515392038603),
 ('occupation_ Armed-Forces', 2.435647824347152e-05),
 ('occupation_ Craft-repair', 0.004138335327320618),
 ('occupation_ Exec-managerial', 0.014679244847574097),
 ('occupation_ Farming-fishing', 0.0030704844933197354),
 ('occupation_ Handlers-cleaners', 0.0025893666800995666),
 ('occupation_ Machine-op-inspct', 0.003056370026968405),
 ('occupation_ Other-service', 0.00714051282609981),
 ('occupation_ Priv-house-serv', 0.00018831902815490088),
 ('occupation_ Prof-specialty', 0.010808715750820859),
 ('occupation_ Protective-serv', 0.0023972092844161564),
 ('occupation_ Sales', 0.004421453447779318),
 ('occupation_ Tech-support', 0.00315711973279361),
 ('occupation_ Transport-moving', 0.0029911612489300346),
 ('workclass', 0.01619216171676331),
 ('education', 0.02165958605785938),
 ('marital-status', 0.054379727522097146),
 ('occupation', 0.025993169328948126)]
```

In [510…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [511…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.839690436705362

Confusion Matrix
[[11318  1117]
 [ 1493  2353]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.88      0.91      0.90     12435
         1.0       0.68      0.61      0.64      3846

    accuracy                           0.84     16281
   macro avg       0.78      0.76      0.77     16281
weighted avg       0.83      0.84      0.84     16281
```

Add Relationship

In [512…

```
transform_columns = ['sex', "workclass", "education", "marital-status", "occupation",
non_num_columns = ["workclass", "education", "marital-status",
                   "occupation", "relationship",
                   'race', 'sex', 'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])
x["marital-status"] = enc.fit_transform(df[["marital-status"]])
x["occupation"] = enc.fit_transform(df[["occupation"]])
x["relationship"] = enc.fit_transform(df[["relationship"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
xt["marital-status"] = enc.fit_transform(golden[["marital-status"]])
xt["occupation"] = enc.fit_transform(golden[["occupation"]])
xt["relationship"] = enc.fit_transform(golden[["relationship"]])
```

In [513…

```
x.head()
```

Out[513]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_Female	sex_Male	workclass_?	workclass_Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, Education, Marital Status, Occupation, and Relationship

In [514…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[514]:
```
[('age', 0.25065656555560434),
 ('education-num', 0.05217618842932025),
 ('capital-gain', 0.1047032181524999),
 ('capital-loss', 0.0334246360356077),
 ('hours-per-week', 0.12620296257073002),
 ('sex_ Female', 0.006584927350715258),
 ('sex_ Male', 0.008810103683552957),
 ('workclass_ ?', 0.001108212081583219),
 ('workclass_ Federal-gov', 0.0038729023419535936),
 ('workclass_ Local-gov', 0.004789117873909931),
 ('workclass_ Never-worked', 8.80718055373483e-06),
 ('workclass_ Private', 0.007931896327330436),
 ('workclass_ Self-emp-inc', 0.004889482910535244),
 ('workclass_ Self-emp-not-inc', 0.006089722247538232),
 ('workclass_ State-gov', 0.0033705429385505097),
 ('workclass_ Without-pay', 7.50956355664963e-05),
 ('education_ 10th', 0.0010218917827485993),
 ('education_ 11th', 0.001253959189808681),
 ('education_ 12th', 0.000568985493818007),
 ('education_ 1st-4th', 0.00019608532606966273),
 ('education_ 5th-6th', 0.0005763866099762815),
 ('education_ 7th-8th', 0.0014682556057896597),
 ('education_ 9th', 0.00085341571476573),
 ('education_ Assoc-acdm', 0.001860993186311075),
 ('education_ Assoc-voc', 0.002403171059093996),
 ('education_ Bachelors', 0.0079870452143942),
 ('education_ Doctorate', 0.002318116056649682),
 ('education_ HS-grad', 0.0072298524431998),
 ('education_ Masters', 0.005081089977736375),
 ('education_ Preschool', 0.0001593931466641087),
 ('education_ Prof-school', 0.003101527460932004),
 ('education_ Some-college', 0.003812209481317725),
 ('marital-status_ Divorced', 0.004349324162476512),
 ('marital-status_ Married-AF-spouse', 0.0002799012370117445),
 ('marital-status_ Married-civ-spouse', 0.06494950763193764),
 ('marital-status_ Married-spouse-absent', 0.000820869876596392),
 ('marital-status_ Never-married', 0.016367429616420726),
 ('marital-status_ Separated', 0.0016085371828317094),
 ('marital-status_ Widowed', 0.0014275427518427402),
 ('occupation_ ?', 0.0009964461329601795),
 ('occupation_ Adm-clerical', 0.0034841136136246585),
 ('occupation_ Armed-Forces', 1.9853706947836274e-05),
 ('occupation_ Craft-repair', 0.004155961102616327 4),
 ('occupation_ Exec-managerial', 0.015390684587209281),
 ('occupation_ Farming-fishing', 0.0029859933944949993),
 ('occupation_ Handlers-cleaners', 0.0023428188742755838),
 ('occupation_ Machine-op-inspct', 0.002874866211642823),
 ('occupation_ Other-service', 0.00584006301826 7305),
 ('occupation_ Priv-house-serv', 0.00012831310530169658),
 ('occupation_ Prof-specialty', 0.011879441113288055),
 ('occupation_ Protective-serv', 0.002143708655816767),
 ('occupation_ Sales', 0.00435663230221897),
 ('occupation_ Tech-support', 0.0032314047891470175),
 ('occupation_ Transport-moving', 0.0029349674920431616),
 ('relationship_ Husband', 0.029702694786659593),
 ('relationship_ Not-in-family', 0.008040038003911005),
 ('relationship_ Other-relative', 0.0013533316408985896),
 ('relationship_ Own-child', 0.00835970435269773 1),
 ('relationship_ Unmarried', 0.0038191096328510465),
 ('relationship_ Wife', 0.00866272102457711),
```

```
('workclass', 0.01620502778356909),
('education', 0.022089779352607964),
('marital-status', 0.021689685440717966),
('occupation', 0.025355186891860292),
('relationship', 0.04756758146585019)]
```

In [515…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [516…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8406731773232602

Confusion Matrix
[[11341  1094]
 [ 1500  2346]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.88      0.91      0.90     12435
         1.0       0.68      0.61      0.64      3846

    accuracy                           0.84     16281
   macro avg       0.78      0.76      0.77     16281
weighted avg       0.84      0.84      0.84     16281
```

Add Race

In [517…

```
transform_columns = ['sex', "workclass", "education", "marital-status", "occupation",
non_num_columns = ["workclass", "education", "marital-status",
                   "occupation", "relationship", "race", 'sex', 'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])
x["marital-status"] = enc.fit_transform(df[["marital-status"]])
x["occupation"] = enc.fit_transform(df[["occupation"]])
x["relationship"] = enc.fit_transform(df[["relationship"]])
x["race"] = enc.fit_transform(df[["race"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
```

```
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
xt["marital-status"] = enc.fit_transform(golden[["marital-status"]])
xt["occupation"] = enc.fit_transform(golden[["occupation"]])
xt["relationship"] = enc.fit_transform(golden[["relationship"]])
xt["race"] = enc.fit_transform(golden[["race"]])
```

In [518…

```
x.head()
```

Out[518]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_Female	sex_Male	workclass_?	workclass_Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, Education, Marital Status, Occupation, Relationship, and Race

In [519…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[519]:
```
[('age', 0.2425227572890333),
 ('education-num', 0.05422198136232316),
 ('capital-gain', 0.09603615853587484),
 ('capital-loss', 0.03158983186737807),
 ('hours-per-week', 0.11924642848028093),
 ('sex_ Female', 0.006005311625463913 5),
 ('sex_ Male', 0.008446231748442447),
 ('workclass_ ?', 0.001184336763724142),
 ('workclass_ Federal-gov', 0.003888642001739736),
 ('workclass_ Local-gov', 0.004765615553060558),
 ('workclass_ Never-worked', 4.15366711949797e-06),
 ('workclass_ Private', 0.00849556410703703),
 ('workclass_ Self-emp-inc', 0.004743294951642238),
 ('workclass_ Self-emp-not-inc', 0.006340330942789592),
 ('workclass_ State-gov', 0.003430275909092427),
 ('workclass_ Without-pay', 4.8698028505016205e-05),
 ('education_ 10th', 0.001140469244414851),
 ('education_ 11th', 0.0013571229735972136),
 ('education_ 12th', 0.0006050923510190408),
 ('education_ 1st-4th', 0.0002221145892936912),
 ('education_ 5th-6th', 0.0006415288671685205),
 ('education_ 7th-8th', 0.0011898283454329342),
 ('education_ 9th', 0.0009638614507189542),
 ('education_ Assoc-acdm', 0.0018589972746047637),
 ('education_ Assoc-voc', 0.002414258629354344),
 ('education_ Bachelors', 0.008236710210799816),
 ('education_ Doctorate', 0.0021656712918622814),
 ('education_ HS-grad', 0.007992707122968417),
 ('education_ Masters', 0.004694659822049571),
 ('education_ Preschool', 0.00012762412175423116),
 ('education_ Prof-school', 0.0032128139609139588),
 ('education_ Some-college', 0.004379847963789885),
 ('marital-status_ Divorced', 0.004754789637714388),
 ('marital-status_ Married-AF-spouse', 0.000373900681519155),
 ('marital-status_ Married-civ-spouse', 0.060631718851543975),
 ('marital-status_ Married-spouse-absent', 0.0007391760694854),
 ('marital-status_ Never-married', 0.020709535573736535),
 ('marital-status_ Separated', 0.0012254878400222778),
 ('marital-status_ Widowed', 0.0012620339137062052),
 ('occupation_ ?', 0.0011806270266988223),
 ('occupation_ Adm-clerical', 0.003928652136977034),
 ('occupation_ Armed-Forces', 1.2534315092337232e-05),
 ('occupation_ Craft-repair', 0.004475655603944 1155),
 ('occupation_ Exec-managerial', 0.014565718023891476),
 ('occupation_ Farming-fishing', 0.003292306087345848),
 ('occupation_ Handlers-cleaners', 0.002514657162978674),
 ('occupation_ Machine-op-inspct', 0.003076821154941127),
 ('occupation_ Other-service', 0.006005135784531002),
 ('occupation_ Priv-house-serv', 0.00014936666320687265),
 ('occupation_ Prof-specialty', 0.01118471552834892),
 ('occupation_ Protective-serv', 0.0024219583858306323),
 ('occupation_ Sales', 0.004835861368174915),
 ('occupation_ Tech-support', 0.003354215514242232),
 ('occupation_ Transport-moving', 0.00318101978619682),
 ('relationship_ Husband', 0.029877135727288437),
 ('relationship_ Not-in-family', 0.005925163156571129),
 ('relationship_ Other-relative', 0.0011121409851091199),
 ('relationship_ Own-child', 0.008157665463140448),
 ('relationship_ Unmarried', 0.0034386477422398806),
 ('relationship_ Wife', 0.008755960885404066),
```

```
('race_ Amer-Indian-Eskimo', 0.0011081777670886718),
('race_ Asian-Pac-Islander', 0.0031816579885002114),
('race_ Black', 0.004616336439692352),
('race_ Other', 0.0010317635118937511),
('race_ White', 0.005573118877478422 6),
('workclass', 0.01643762923275296),
('education', 0.022801467477436808),
('marital-status', 0.02868445270446598),
('occupation', 0.025665734096368905),
('relationship', 0.03959365304173292),
('race', 0.007990486739458003)]
```

In [520…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [521…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8417173392297771

Confusion Matrix
[[11355  1080]
 [ 1497  2349]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.88      0.91      0.90     12435
         1.0       0.69      0.61      0.65      3846

    accuracy                           0.84     16281
   macro avg       0.78      0.76      0.77     16281
weighted avg       0.84      0.84      0.84     16281
```

Add Native Country

In [522…

```
from sklearn import preprocessing
enc = preprocessing.OrdinalEncoder()
```

In [523…

```
transform_columns = ['sex', "workclass", "education", "marital-status",
                     "occupation", "relationship", "race", "native-country"]
non_num_columns = ["workclass", "education", "marital-status",
                   "occupation", "relationship", "race", 'sex', 'native-country']

# Training set
x = df.copy()
x = pd.concat([x.drop(non_num_columns, axis=1),
               pd.get_dummies(df[transform_columns])], axis=1,)
x["salary"] = enc.fit_transform(df[["salary"]])
```

```
x["workclass"] = enc.fit_transform(df[["workclass"]])
x["education"] = enc.fit_transform(df[["education"]])
x["marital-status"] = enc.fit_transform(df[["marital-status"]])
x["occupation"] = enc.fit_transform(df[["occupation"]])
x["relationship"] = enc.fit_transform(df[["relationship"]])
x["race"] = enc.fit_transform(df[["race"]])
x["native-country"] = enc.fit_transform(df[["native-country"]])

#Test set
xt = golden.copy()
xt = pd.concat([xt.drop(non_num_columns, axis=1),
               pd.get_dummies(golden[transform_columns])], axis=1,)
xt["salary"] = enc.fit_transform(golden[["salary"]])
xt["workclass"] = enc.fit_transform(golden[["workclass"]])
xt["education"] = enc.fit_transform(golden[["education"]])
xt["marital-status"] = enc.fit_transform(golden[["marital-status"]])
xt["occupation"] = enc.fit_transform(golden[["occupation"]])
xt["relationship"] = enc.fit_transform(golden[["relationship"]])
xt["race"] = enc.fit_transform(golden[["race"]])
xt["native-country"] = enc.fit_transform(golden[["native-country"]])
```

In [524…

```
x.shape
```

Out[524]:

```
(32561, 116)
```

In [525…

```
# Which column is all zeroes?
pd.set_option("display.max_rows", None, "display.max_columns", None)
print((x == 0).sum(axis = 0))
```

```
age                                         0
fnlwgt                                      0
education-num                               0
capital-gain                            29849
capital-loss                            31042
hours-per-week                              0
salary                                  24720
sex_ Female                             21790
sex_ Male                               10771
workclass_ ?                            30725
workclass_ Federal-gov                  31601
workclass_ Local-gov                    30468
workclass_ Never-worked                 32554
workclass_ Private                       9865
workclass_ Self-emp-inc                 31445
workclass_ Self-emp-not-inc             30020
workclass_ State-gov                    31263
workclass_ Without-pay                  32547
education_ 10th                         31628
education_ 11th                         31386
education_ 12th                         32128
education_ 1st-4th                      32393
education_ 5th-6th                      32228
education_ 7th-8th                      31915
education_ 9th                          32047
education_ Assoc-acdm                   31494
education_ Assoc-voc                    31179
education_ Bachelors                    27206
education_ Doctorate                    32148
education_ HS-grad                      22060
education_ Masters                      30838
education_ Preschool                    32510
education_ Prof-school                  31985
education_ Some-college                 25270
marital-status_ Divorced                28118
marital-status_ Married-AF-spouse       32538
marital-status_ Married-civ-spouse      17585
marital-status_ Married-spouse-absent   32143
marital-status_ Never-married           21878
marital-status_ Separated               31536
marital-status_ Widowed                 31568
occupation_ ?                           30718
occupation_ Adm-clerical                28791
occupation_ Armed-Forces                32552
occupation_ Craft-repair                28462
occupation_ Exec-managerial             28495
occupation_ Farming-fishing             31567
occupation_ Handlers-cleaners           31191
occupation_ Machine-op-inspct           30559
occupation_ Other-service               29266
occupation_ Priv-house-serv             32412
occupation_ Prof-specialty              28421
occupation_ Protective-serv             31912
occupation_ Sales                       28911
occupation_ Tech-support                31633
occupation_ Transport-moving            30964
relationship_ Husband                   19368
relationship_ Not-in-family             24256
relationship_ Other-relative            31580
relationship_ Own-child                 27493
```

```
relationship_ Unmarried                          29115
relationship_ Wife                               30993
race_ Amer-Indian-Eskimo                         32250
race_ Asian-Pac-Islander                         31522
race_ Black                                      29437
race_ Other                                      32290
race_ White                                       4745
native-country_ ?                                31978
native-country_ Cambodia                         32542
native-country_ Canada                           32440
native-country_ China                            32486
native-country_ Columbia                         32502
native-country_ Cuba                             32466
native-country_ Dominican-Republic               32491
native-country_ Ecuador                          32533
native-country_ El-Salvador                      32455
native-country_ England                          32471
native-country_ France                           32532
native-country_ Germany                          32424
native-country_ Greece                           32532
native-country_ Guatemala                        32497
native-country_ Haiti                            32517
native-country_ Holand-Netherlands               32560
native-country_ Honduras                         32548
native-country_ Hong                             32541
native-country_ Hungary                          32548
native-country_ India                            32461
native-country_ Iran                             32518
native-country_ Ireland                          32537
native-country_ Italy                            32488
native-country_ Jamaica                          32480
native-country_ Japan                            32499
native-country_ Laos                             32543
native-country_ Mexico                           31918
native-country_ Nicaragua                        32527
native-country_ Outlying-US(Guam-USVI-etc)       32547
native-country_ Peru                             32530
native-country_ Philippines                      32363
native-country_ Poland                           32501
native-country_ Portugal                         32524
native-country_ Puerto-Rico                      32447
native-country_ Scotland                         32549
native-country_ South                            32481
native-country_ Taiwan                           32510
native-country_ Thailand                         32543
native-country_ Trinadad&Tobago                  32542
native-country_ United-States                     3391
native-country_ Vietnam                          32494
native-country_ Yugoslavia                       32545
workclass                                         1836
education                                          933
marital-status                                    4443
occupation                                        1843
relationship                                     13193
race                                               311
native-country                                     583
dtype: int64
```

In [526…

```
# Drop Holand-Netherlands
```

```
x.drop("native-country_ Holand-Netherlands", axis = 1, inplace = True)
```

In [527…

```
x.head()
```

Out[527]:

	age	fnlwgt	education-num	capital-gain	capital-loss	hours-per-week	salary	sex_ Female	sex_ Male	workclass_ ?	workclass_ Federal-gov
0	39	77516	13	2174	0	40	0.0	0	1	0	0
1	50	83311	13	0	0	13	0.0	0	1	0	0
2	38	215646	9	0	0	40	0.0	0	1	0	0
3	53	234721	7	0	0	40	0.0	0	1	0	0
4	28	338409	13	0	0	40	0.0	1	0	0	0

Random Forest with Sex, Workclass, Education, Marital Status, Occupation, Relationship, Race, and Native Country

In [528…

```
model = RandomForestClassifier(criterion='entropy')
model.fit(x.drop(['fnlwgt','salary'], axis=1), x.salary)
list(zip(x.drop(['fnlwgt','salary'], axis=1).columns, model.feature_importances_))
```

Out[528]:
```
[('age', 0.22911851944161704),
 ('education-num', 0.046651299593324175),
 ('capital-gain', 0.09331447269435941),
 ('capital-loss', 0.030160874174879507),
 ('hours-per-week', 0.1144725252076264 2),
 ('sex_ Female', 0.007169381912918505),
 ('sex_ Male', 0.01011092944849207),
 ('workclass_ ?', 0.0011597640935645012),
 ('workclass_ Federal-gov', 0.0037692789501626084),
 ('workclass_ Local-gov', 0.004801953128434847),
 ('workclass_ Never-worked', 5.439673478955258e-06),
 ('workclass_ Private', 0.008167911265165823),
 ('workclass_ Self-emp-inc', 0.004517169196345229),
 ('workclass_ Self-emp-not-inc', 0.005938050240134955),
 ('workclass_ State-gov', 0.0034002613308722366),
 ('workclass_ Without-pay', 6.351162786161508e-05),
 ('education_ 10th', 0.0011808476995083562),
 ('education_ 11th', 0.0014195649553995916),
 ('education_ 12th', 0.000708225711277857),
 ('education_ 1st-4th', 0.0002516797077795117),
 ('education_ 5th-6th', 0.00046880122800401066),
 ('education_ 7th-8th', 0.0013780605197210392),
 ('education_ 9th', 0.0009102770290536573),
 ('education_ Assoc-acdm', 0.0020491095043190075),
 ('education_ Assoc-voc', 0.0024725813389284133),
 ('education_ Bachelors', 0.008274984240637646),
 ('education_ Doctorate', 0.002404264109301696),
 ('education_ HS-grad', 0.007741115599534689),
 ('education_ Masters', 0.006291502333164281),
 ('education_ Preschool', 9.073206540235364e-05),
 ('education_ Prof-school', 0.003238691098867652),
 ('education_ Some-college', 0.004520123396596785),
 ('marital-status_ Divorced', 0.004292590960627863),
 ('marital-status_ Married-AF-spouse', 0.0002654970641659582),
 ('marital-status_ Married-civ-spouse', 0.057130881323937205),
 ('marital-status_ Married-spouse-absent', 0.0007685251567880951),
 ('marital-status_ Never-married', 0.022132637767061057),
 ('marital-status_ Separated', 0.001443977997274856),
 ('marital-status_ Widowed', 0.0012920656150600617),
 ('occupation_ ?', 0.0012570658092493742),
 ('occupation_ Adm-clerical', 0.003888183402377345),
 ('occupation_ Armed-Forces', 7.849021305183289e-06),
 ('occupation_ Craft-repair', 0.004594382945148611 4),
 ('occupation_ Exec-managerial', 0.014274208251001674),
 ('occupation_ Farming-fishing', 0.0031239816488923783),
 ('occupation_ Handlers-cleaners', 0.0026827428012387677),
 ('occupation_ Machine-op-inspct', 0.003395624395002198),
 ('occupation_ Other-service', 0.006374929995402671),
 ('occupation_ Priv-house-serv', 0.00013295517455328977),
 ('occupation_ Prof-specialty', 0.011420136490658232),
 ('occupation_ Protective-serv', 0.0021654014397871086),
 ('occupation_ Sales', 0.004646477038827562),
 ('occupation_ Tech-support', 0.003019921089678115),
 ('occupation_ Transport-moving', 0.0031902109787092844),
 ('relationship_ Husband', 0.03021692906439417),
 ('relationship_ Not-in-family', 0.008853978064457679),
 ('relationship_ Other-relative', 0.0012053646136405563),
 ('relationship_ Own-child', 0.006157719356010141),
 ('relationship_ Unmarried', 0.003877349467876043),
 ('relationship_ Wife', 0.007540987901406903),
```

```
('race_ Amer-Indian-Eskimo', 0.0010783387378159985),
('race_ Asian-Pac-Islander', 0.0024417344992135634),
('race_ Black', 0.004280975301823648),
('race_ Other', 0.0007538656099764076),
('race_ White', 0.005175528182843353),
('native-country_ ?', 0.0019910876186786685),
('native-country_ Cambodia', 0.00019533004154127169),
('native-country_ Canada', 0.0008951951867414616),
('native-country_ China', 0.0003895201639558183),
('native-country_ Columbia', 0.00018952273666699625),
('native-country_ Cuba', 0.0006145896047382923),
('native-country_ Dominican-Republic', 0.00013716177203916308),
('native-country_ Ecuador', 0.00012534720596171515),
('native-country_ El-Salvador', 0.00020961519482565817),
('native-country_ England', 0.0007941549505858154),
('native-country_ France', 0.0002532304006155029),
('native-country_ Germany', 0.0008619788400652217),
('native-country_ Greece', 0.00027927028479582167),
('native-country_ Guatemala', 0.00016610100065482167),
('native-country_ Haiti', 0.000154256795821712),
('native-country_ Honduras', 6.4998998750552595e-06),
('native-country_ Hong', 0.00010302522226291227),
('native-country_ Hungary', 0.00011605434120539044),
('native-country_ India', 0.0006563692149357495),
('native-country_ Iran', 0.0003724525770610461),
('native-country_ Ireland', 0.00018482670535046146),
('native-country_ Italy', 0.0006919008152669361),
('native-country_ Jamaica', 0.00031122003121692213),
('native-country_ Japan', 0.0004900649797857251),
('native-country_ Laos', 9.244800802157836e-05),
('native-country_ Mexico', 0.0017742723163423266),
('native-country_ Nicaragua', 0.0001341615459447685),
('native-country_ Outlying-US(Guam-USVI-etc)', 2.2204394964740045e-05),
('native-country_ Peru', 8.937308063042027e-05),
('native-country_ Philippines', 0.0008072272852147244),
('native-country_ Poland', 0.0004977404449894836),
('native-country_ Portugal', 0.00018845547488740118),
('native-country_ Puerto-Rico', 0.0005005853553712369),
('native-country_ Scotland', 8.193847573632375e-05),
('native-country_ South', 0.0005340837095054746),
('native-country_ Taiwan', 0.00023361339195195806),
('native-country_ Thailand', 6.53684492277114e-05),
('native-country_ Trinadad&Tobago', 8.913471061092276e-05),
('native-country_ United-States', 0.004847670971380884),
('native-country_ Vietnam', 0.00029283778516370906),
('native-country_ Yugoslavia', 0.000221205932593295),
('workclass', 0.015787300890446332),
('education', 0.02413550623131719),
('marital-status', 0.03839544240797084),
('occupation', 0.02585307578211641),
('relationship', 0.030100123212637043),
('race', 0.006960941765454761),
('native-country', 0.008801593109932561)]
```

In [529…

```
predictions = model.predict(xt.drop(['fnlwgt','salary'], axis=1))
```

In [530…

```
print("Accuracy Score")
print(accuracy_score(xt.salary, predictions))
```

```
print("")
print("Confusion Matrix")
print(confusion_matrix(xt.salary, predictions))
print("")
print("Classification Report")
print(classification_report(xt.salary, predictions))
```

```
Accuracy Score
0.8413488114980652

Confusion Matrix
[[11397  1038]
 [ 1545  2301]]

Classification Report
              precision    recall  f1-score   support

         0.0       0.88      0.92      0.90     12435
         1.0       0.69      0.60      0.64      3846

    accuracy                           0.84     16281
   macro avg       0.78      0.76      0.77     16281
weighted avg       0.84      0.84      0.84     16281
```

In [ ]: