AI Final project
Income Classification

COMP 3710

Applied Artificial Intelligence

Grant Marshall

T00569969

Introduction

The purpose of this study was to investigate the Adult Census Income (American census data) and do EDA (exploratory data analysis) and build a 'model' for predicting which category a data point fits into based on mostly numerical factors.

Some questions asked were:

- How does the number of years in education correlate to probability of making more than 50k?
- Does marital status affect it?
- What is the variety/diversity of the data set
- Does private do better than public jobs?

The first step of the process is to import and process the data using mainly pandas and numpy. This process includes filling null values with numpy's nan values for simplicity and processing then removing '?' characters used in the dataset as they serve no purpose. I then filtered and encoded important categorical values into simple numerical components for the machine learning algorithms. I mapped all values describing income above 50k (>50k,>50k.) as 1 and less than or equal(<=50k,<=50k.) as 0. Occupation and workclass had many nulls that needed to be filled. The final steps were to map Male to 0 and Female to 1 and build the train and test data sets using scikit-learn and pandas to drop unrelated columns and separate income from the test set.

Background

Plenty of work has been done with this dataset as it has been well known as a good beginner dataset for classification. I used a process similar to one-hot encoding to convert categorical variables to numerical for use in the classification task. The concept of feature engineering and data cleaning is used throughout to provide a clean dataset and to focus on the data being examined. I am using cross validation to test the models for accuracy and validate the results adcross samples. I am using Logistic Regression, K Nearest Neighbors,Decision Tree, Naive Bayes, Random Forests, and Support Vector Machine classifiers. **Important note:** I am using **scikit 0.19.2** vs the latest 0.21 as I was having issues with SVC otherwise. I intended to include marital status into my model but I was having errors including it so it was scrapped.

Details

```
import matplotlib.pyplot as plt # graphing lib
import numpy as np # linear alg.
import pandas as pd # data manipulation
import seaborn as sns # nicer matplot graphs
%matplotlib inline # inline graphs are nice to have

# name columns, some preprocessing was done in excel to switch
from marital.status to marital_status and removed the original
headers which were in line 0/1 of the csv
name_headers =
['age','workclass','fnlwgt','education','education_num',
          'marital_status','occupation',
          'relationship','race','sex','capital_gain',

'capital_loss','hours_per_week','native_country','income']
# read data in
df=pd.read_csv('adult.csv',names=name_headers)
```

This section of code import the current relevant libraries and pulls in the csv file from Kaggle with pandas headers for easy addressing.

```
df = df.fillna(np.nan)
df = df.applymap(lambda x: np.nan if x=='?' else x)
sns.set(color_codes=True)
```

Apply np.nan to fields already provided as null, remove '?'s and set seaborn color code config.

```
#looking at age distribution
#primarily middle age majority 20-50s
#df['age'].value_counts()
sns.distplot(df['age'],bins=8, kde=False) # kde=False to remove
line plot.
```

I found 8 bins to be a good number to capture the ages properly for this data.

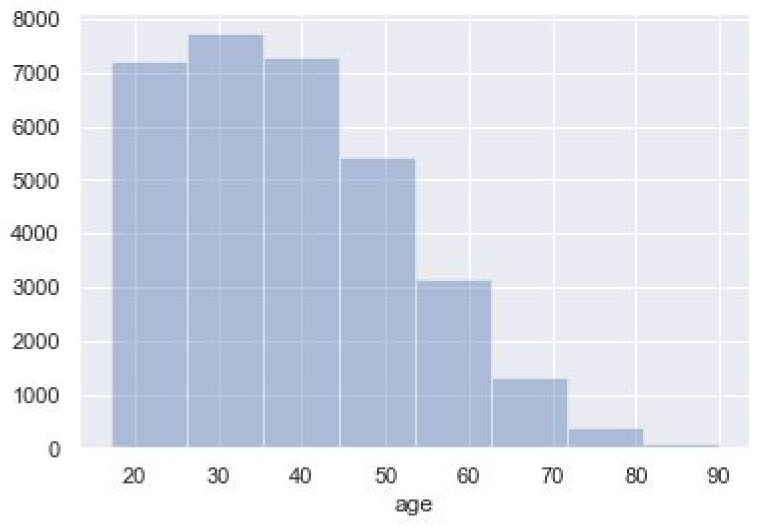


Figure 1: Age distribution

```
# see race dist, primarily white
df['race'].value_counts()
sns.countplot(df['race'])
```

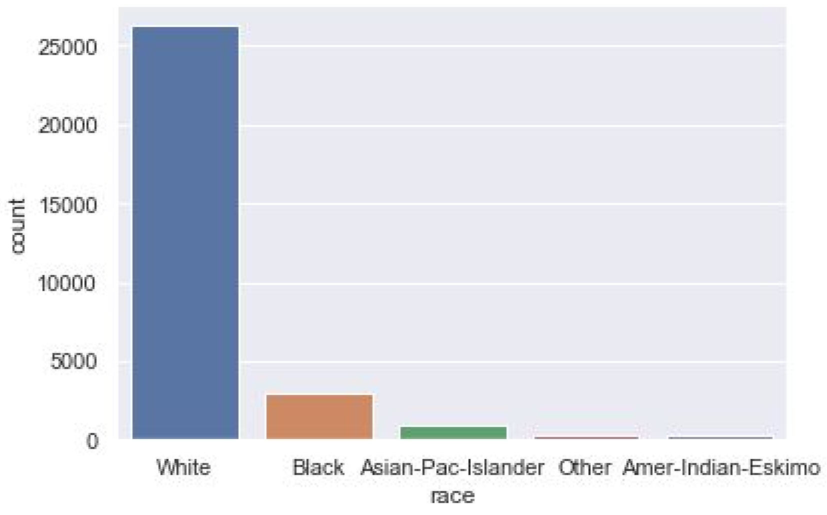


Figure 2: Race distribution, this set is predominantly White Americans

```
# primarily private sector
df['workclass'].value_counts()
count = sns.countplot(df['workclass'])
# set x axis size later
```

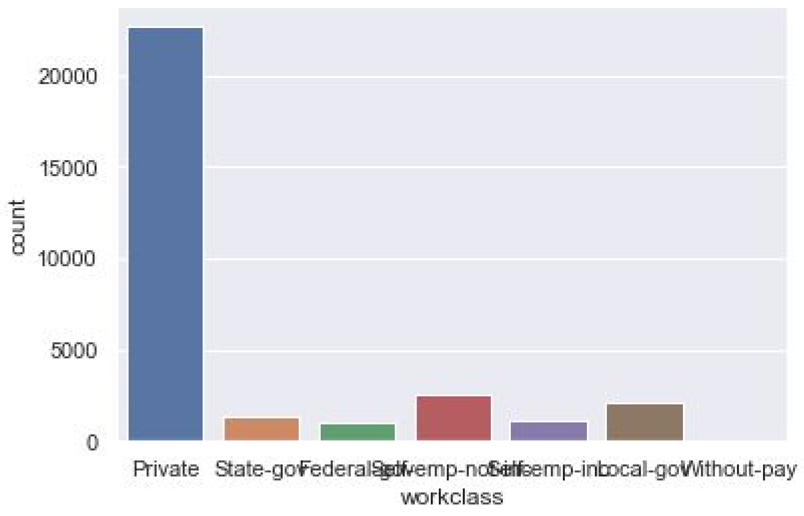


Figure 3: Workclass distribution

```
plt.figure(figsize=(10,5))
chart = sns.countplot(df['marital_status'])
chart.set_xticklabels(chart.get_xticklabels(),rotation=45) #
proper x labels now!!
```

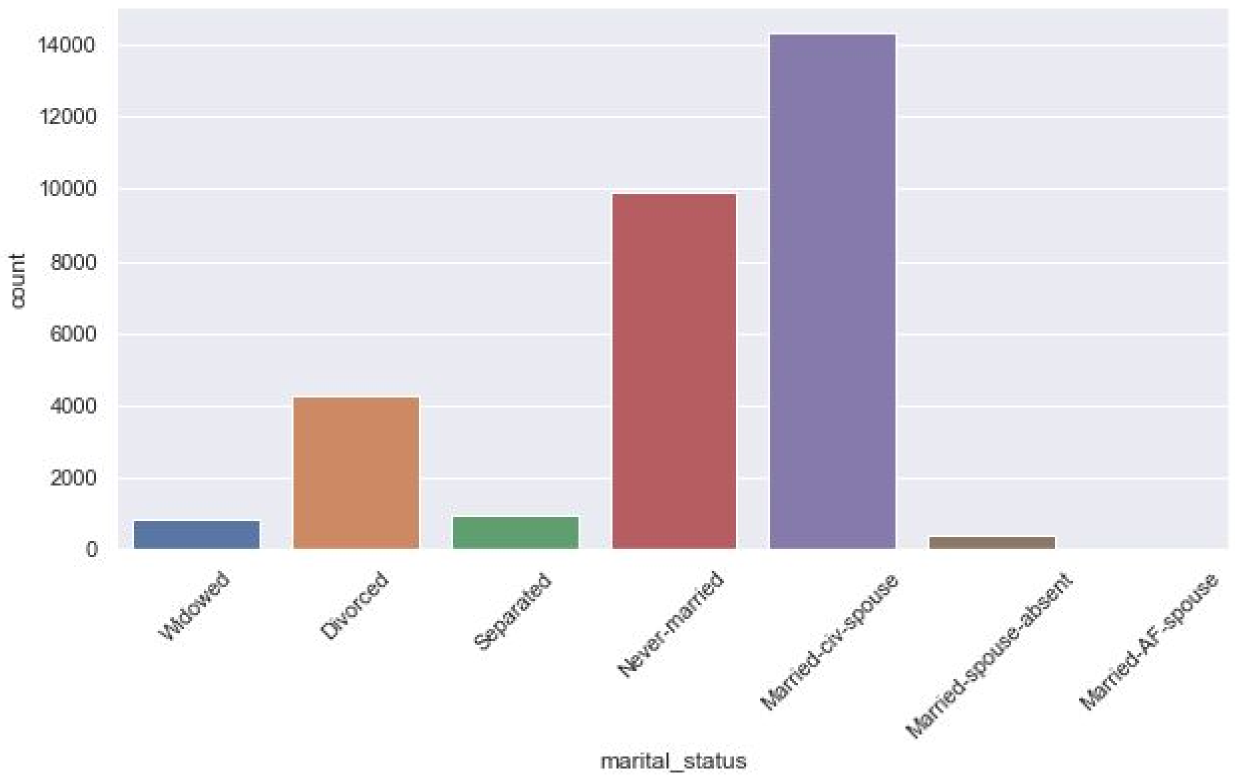


Figure 4: Marital Status distribution

```
sns.countplot(df['sex'])
```

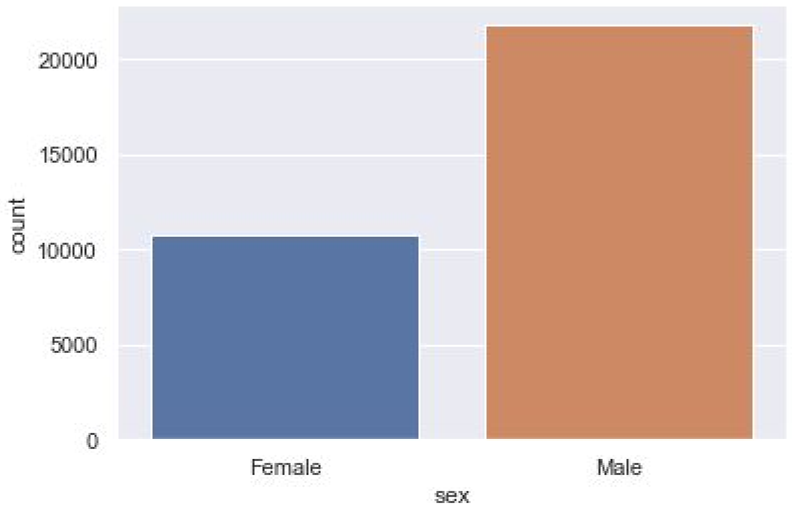


Figure 5: Distribution of Male and Female

So, so far Predominantly White Male American in middle age. Not too representative of everyone really and as shown below mostly lower than 50k.

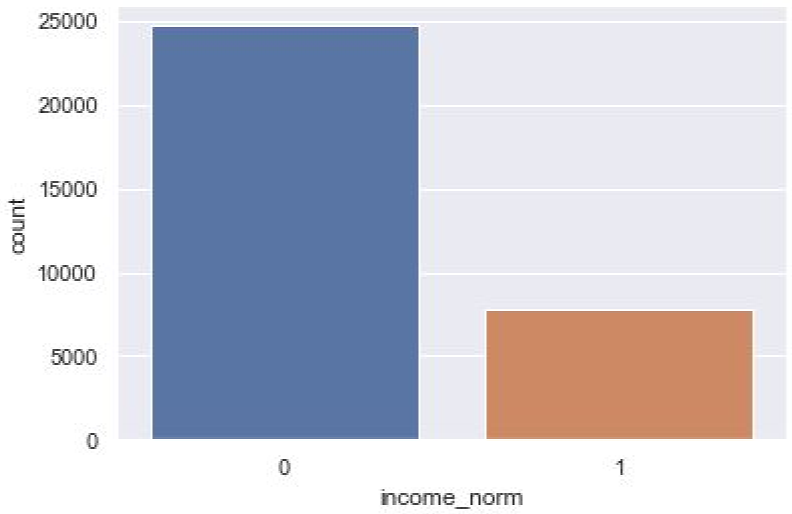
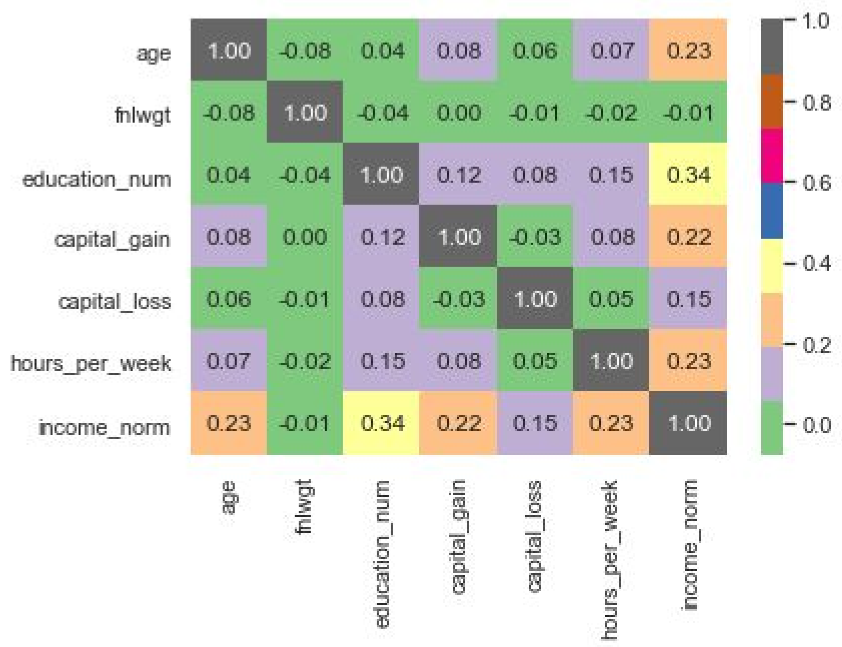


Figure 6: incorm normalized to 0,1 as seen below in the code.

```
sns.heatmap(df[numeric_features].corr(),
fmt='.2f',cmap='Accent',annot=True)
```



	age	fnlwgt	education_num	capital_gain	capital_loss	hours_per_week	income_norm
age	1.00	-0.08	0.04	0.08	0.06	0.07	0.23
fnlwgt	-0.08	1.00	-0.04	0.00	-0.01	-0.02	-0.01
education_num	0.04	-0.04	1.00	0.12	0.08	0.15	0.34
capital_gain	0.08	0.00	0.12	1.00	-0.03	0.08	0.22
capital_loss	0.06	-0.01	0.08	-0.03	1.00	0.05	0.15
hours_per_week	0.07	-0.02	0.15	0.08	0.05	1.00	0.23
income_norm	0.23	-0.01	0.34	0.22	0.15	0.23	1.00

Figure 7: Correlation Heat Map

```
from sklearn.model_selection import
train_test_split,cross_val_score # split data and validate
models

#map that we want to predict to a binary set of ints (0,1)
df['income_norm']=df['income'].map({'<=50K': 0, '>50K': 1,
'<=50K.': 0, '>50K.': 1})
#Numeric features
numeric_features =
['age','fnlwgt','education_num','capital_gain','capital_loss','h
ours_per_week','income_norm']
#Categorical features
cat_features = ['workclass','education','marital_status',
'occupation', 'relationship', 'race', 'sex', 'native']
# data cleaning
df['workclass'] = df['workclass'].fillna('X')
df['occupation'] = df['occupation'].fillna('X')
```

```
df['native_country'] =
df['native_country'].fillna('United-States')
#check to be sure we have no na
df.isnull().sum()
df.drop(labels=['education','occupation','workclass','fnlwgt','r
elationship','native_country','income','race','marital_status'],
axis=1, inplace=True)
# income is dropped as incorm_norm is used ^
```

Actual ML Work:

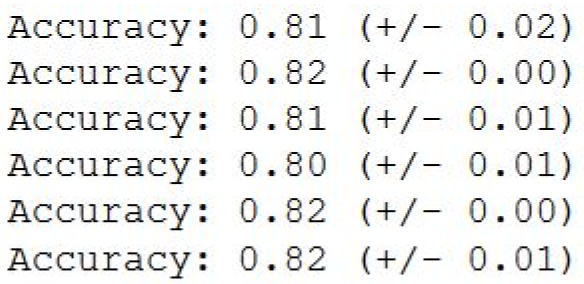
```
df = df.dropna()
y = df['income_norm'].values
x = df.drop(labels=['income_norm'] ,axis=1) # income is the
target
#df.isnull().sum()
# data splitting
X_train,X_test, Y_train, Y_test =
train_test_split(x,y,test_size=0.2,train_size=0.8,random_state =
42)

# sklearn imports
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC

classifiers = []
classifiers.append(LogisticRegression(solver='lbfgs'))
classifiers.append(KNeighborsClassifier())
classifiers.append(DecisionTreeClassifier())
classifiers.append(GaussianNB())
classifiers.append(RandomForestClassifier(n_estimators=100,max_f
eatures=3))
classifiers.append(SVC())
```

```
results=[]
# try each classifier using cross validation scoring and append
for later analysis
for clf in classifiers:
        cross_val  =
cross_val_score(clf,X_train,Y_train,scoring='accuracy',cv=5,n_jobs=-1)
        results.append(cross_val)
        #print("Mean: ",cross_val.mean(),"Standard Dev: ",cross_val.std()*2)
        print("Accuracy: %0.2f (+/- %0.2f)" % (cross_val.mean(), cross_val.std() * 2))
```

Results
Models return roughly 80% accuracy which is acceptable but could be improved in future.



```
Accuracy: 0.81 (+/- 0.02)
Accuracy: 0.82 (+/- 0.00)
Accuracy: 0.81 (+/- 0.01)
Accuracy: 0.80 (+/- 0.01)
Accuracy: 0.82 (+/- 0.00)
Accuracy: 0.82 (+/- 0.01)
```

Figure 8: Results of Classifiers on dataset. Generated by code above.
From top down: Logistic Regression,kNN,Decision Tree,Gaussian Naive Bayes, Random Forest,Support Vector Classifier.

Most of the results of my proding have been sprinkled into the details section.

Discussion & Conclusion

Now that this project has come to a close, I am happy with these results as 80% is acceptable given my knowledge of the matter and time, dataset. In Figure 7 the correlation plot. The most correlated variables are the numerical ones with age and income, hours worked and income etc. nothing seemed highly correlated and the results were as expected. Education years and income is expected as well or at least as we expect in society. This dataset however is likely older data due to privacy by the US Census Bureau and is skewed to one prominent profile White Male American making less than 50k USD at the time of this census (which could be inflated to much more now).

Future Plans

If I had a continuous value for income I would do some more Regression to predict a continuous variable for Income as a opposed to a Binary Classification. I would try to fit more variables into my project and test more classifiers to see if it could be improved that way. Possible parameter tuning could be used also boost efficiency. I also could strip income and try clustering to see what patterns form in the data.

References (non Exhaustive unfortunately):

3.1. Cross-validation: evaluating estimator performance¶. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/cross_validation.html.

How to rotate axis labels in seaborn and matplotlib. (n.d.). Retrieved November 29, 2019, from https://www.drawingfromdata.com/how-to-rotate-axis-labels-in-seaborn-and-matplotlib.

Nelson, D. (2019, May 11). Overview of Classification Methods in Python with Scikit-Learn. Retrieved from https://stackabuse.com/overview-of-classification-methods-in-python-with-scikit-learn/.

Uci. (2016, October 7). Adult Census Income. Retrieved from https://www.kaggle.com/uciml/adult-census-income.

What is One Hot Encoding? Why And When do you have to use it? (n.d.). Retrieved from https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f.

Various Kaggle Notebooks for inspirations or pieces