Introduction

For the income prediction project, I've completed data cleaning, data pre-processing and feature exploration. Besides, I've applied two basic models: logistic regression and decision tree without fine-tuning parameters or cross validation.

Data Cleaning

```
train.head()
```

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss	hours.per.week	nati
0	53	Self-emp-not-inc	93449	Prof-school	15	Married-civ-spouse	Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	
1	33	Self-emp-not-inc	123424	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	40	U
2	47	Private	144844	HS-grad	9	Married-civ-spouse	Craft-repair	Husband	White	Male	0	0	40	U
3	40	Private	114580	HS-grad	9	Divorced	Craft-repair	Other-relative	White	Female	0	0	40	
4	39	Private	115618	HS-grad	9	Married-civ-spouse	Transport-moving	Husband	White	Male	0	0	50	U

Noticing that there is a duplicated column 'education' which is in nonnumerical format, we can drop it.

Now we have the matching education level as below:

The matching education level of the education number:

1: Preschool, 2: 1st-4th, 3: 5th-6th, 4: 7th-8th, 5: 9th, 6: 10th, 7: 11th, 8: 12th, 9: HS-grad, 10: Some-college, 11: Assoc-voc, 12: Assoc-acdm, 13: Bachelors, 14: Masters, 15: Prof-school, 16: Doctorate.

Next, I checked for null values, and it appears that there are no nulls in the whole dataset.

Then I checked duplicates and remove them:

```
# check for duplicates
print("Train data:")
print("Before removing duplicates:", train.duplicated().sum())

train = train[~train.duplicated()]

print("After removing duplicates:", train.duplicated().sum())

print("Test data:")
print("Before removing duplicates:", test.duplicated().sum())

test = test[~test.duplicated()]

print("After removing duplicates:", test.duplicated().sum())
```

```
Train data:
Before removing duplicates: 14
After removing duplicates: 0
Test data:
Before removing duplicates: 0
After removing duplicates: 0
```

The next step is to remove the spaces in all data entries for easier access.

```
olumns = ['workclass', 'marital.status', 'occupation', 'relationship', 'race', 'sex', 'native.country']
'or column in columns:
    train[column] = train[column].str.strip()
    test[column] = test[column].str.strip()
```

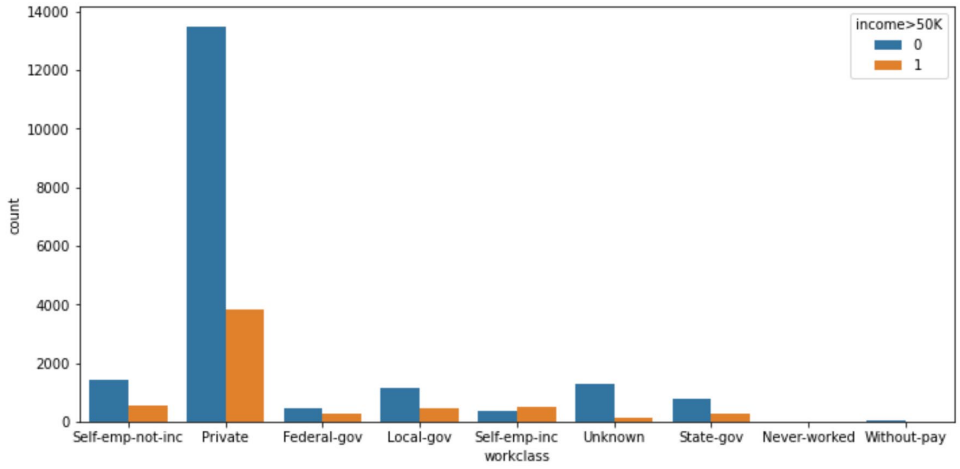
I noticed that there are some '?' symbols in data entries so I decided to change '?' to category 'Unknown'.

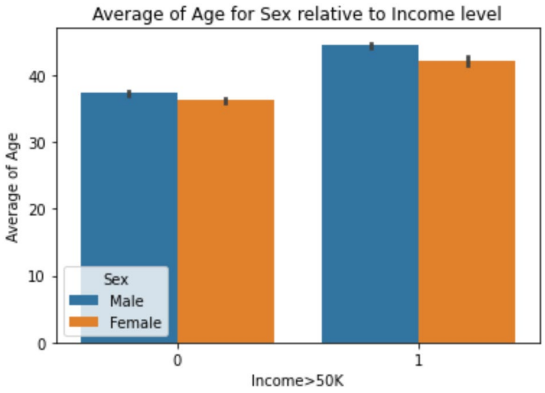
```
train['occupation'].value_counts()
```

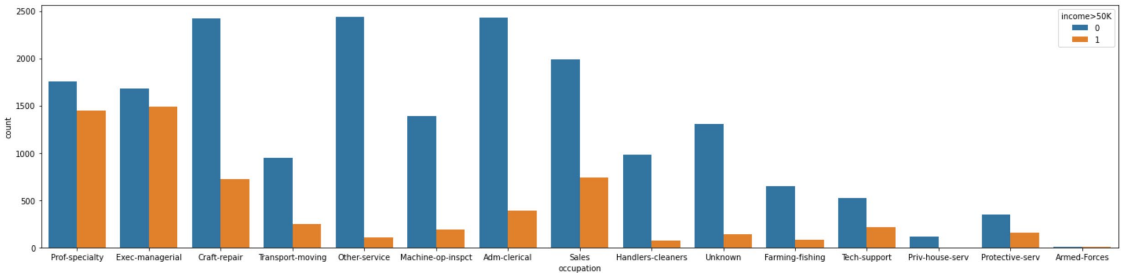
```
Prof-specialty       3202
Exec-managerial      3171
Craft-repair         3144
Adm-clerical         2826
Sales                2727
Other-service        2546
Machine-op-inspct    1580
?                    1441
Transport-moving     1192
Handlers-cleaners    1047
Tech-support          743
Farming-fishing       733
Protective-serv       506
Priv-house-serv       118
Armed-Forces           10
Name: occupation, dtype: int64
```

Feature Exploration

Now the datasets have been prepared and I started feature exploration. In mid-term stage, I only explored a few features: workclass, age, sex and occupation.







Looks like people with >50K has a higher average age than the ones with <=50K. And in both cases of income, we see that the male category has a little bit greater age average than the female category. People working in prof-specialty and exec-managerial industries have higher income.

Basic models

At this stage, I did not do scaling for numerical features. After one-hot encoding, I noticed that there are some unseen categorical values in test data, so we need more pre-processing work. I have submitted the prediction results using Logistic regression and Decision Tree model.

```
]: # get cat_columns
cat_columns = ["workclass", "marital.status","occupation","relationship","race","native.country"]
```

```
]: train_prep = pd.get_dummies(train_prep, prefix_sep="__",
                              columns=cat_columns)
```

```
]: cat_dummies = [col for col in train_prep
                 if "__" in col
                 and col.split("__")[0] in cat_columns]
processed_columns = list(train_prep.columns[:])
```

```
]: test_prep = pd.get_dummies(test_prep, prefix_sep="__",
                                 columns=cat_columns)
```

```
]: # remove additional columns
for col in test_prep:
    if ("__" in col) and (col.split("__")[0] in cat_columns) and col not in cat_dummies:
        print("Removing additional feature {}".format(col))
        test_prep.drop(col, axis=1, inplace=True)
```

```
Removing additional feature native.country__Holand-Netherlands
```

```
]: # Add missing columns
for col in cat_dummies:
    if col not in test_prep:
        print("Adding missing feature {}".format(col))
        test_prep[col] = 0
```

Future Schedule

My following plan is to do more feature engineering work, to explore the feature importance level. I will also perform transformations on features that are highly skewed and perform some scaling on numerical features. Normalization ensures that each feature is treated equally when applying supervised learner. I will test more models like Adaboost, Random Forest, Gradient Boosting, SVM and neural network.