Gender Pay Gap

Vishal Gondi, Neha Mathews, Anushka Pandya

Project Goal

The project goal is to analyze the disparity in pay between males and females by using different datasets and examining whether there is bias.

Cleaning the Data

Dataset 1 (Glassdoor Dataset):

[10]:

	JobTitle	Gender	Age	PerfEval	Education	Dept	Seniority	BasePay
0	Graphic Designer	1	18	5	1	Operations	2	42363
1	Software Engineer	0	21	5	1	Management	5	108476
2	Warehouse Associate	1	19	4	3	Administration	5	90208
3	Software Engineer	0	20	5	2	Sales	4	108080
4	Graphic Designer	0	26	5	2	Engineering	5	99464
...	...	...	...	...	...	...	...	...
995	Marketing Associate	1	61	1	0	Administration	1	62644
996	Data Scientist	0	57	1	2	Sales	2	108977
997	Financial Analyst	0	48	1	0	Operations	1	92347
998	Financial Analyst	0	65	2	0	Administration	1	97376
999	Financial Analyst	0	60	1	3	Sales	2	123108

1000 rows × 8 columns

Dataset 2:

[11]:

	sex	age	annhrs	annincome	degree	explevel
0	1	34	1600	10000.0	1.0	12.0
1	1	32	520	9095.0	0.0	14.0
2	1	64	2550	45200.0	0.0	39.0
3	1	50	3072	25000.0	0.0	30.0
4	1	26	2100	24500.0	0.0	8.0
...	...	...	...	...	...	...
10394	1	25	1610	13000.0	0.0	7.0
10395	1	45	1471	19000.0	0.0	21.0
10396	1	32	1072	8500.0	0.0	14.0
10397	1	41	2500	32000.0	0.0	22.0
10398	1	35	2360	27500.0	1.0	17.0

Code for changing categorical variables to numerical:

```
newGender = {'Gender':{'Male':0,'Female':1}}
df = df.replace(newGender)
newEducation = {'Education':{'High School':0,'College':1,'Masters':2,'PhD':3}}
df = df.replace(newEducation)
```

Linear Regression Predictive Model

Dataset 1:

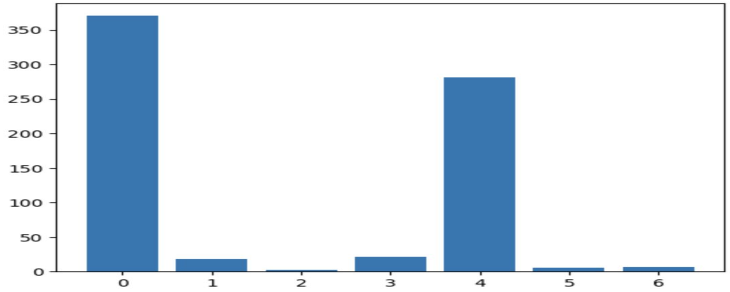
```
train, test = train_test_split(df, test_size=0.2, random_state=1)
model = LinearRegression()
model.fit(
    X=train[["Age", "Gender", "PerfEval", "Education", "Seniority", "JobTitle", "Dept"]],
    y=train["BasePay"]
)
predicted = model.predict(
    X=test[["Age", "Gender", "PerfEval", "Education", "Seniority", "JobTitle", "Dept"]]
)
expected = test["BasePay"]
```

	JobTitle	Gender	Age	PerfEval	Education	Dept	Seniority	BasePay	Bonus	Predicted BasePay Linear
0	0	1	18	5	1	0	2	42363	9938	48713.255227
1	1	0	21	5	1	1	5	108476	11128	91191.043229
2	2	1	19	4	3	2	5	90208	9268	87867.125095
3	1	0	20	5	2	3	4	108080	10154	85603.672625
4	0	0	26	5	2	4	5	99464	9319	101417.038589

```
# feature selection
def select_features(X_train, y_train, X_test):
 # configure to select all features
 fs = SelectKBest(score_func=f_regression, k='all')
 # learn relationship from training data
 fs.fit(X_train, y_train)
 # transform train input data
 X_train_fs = fs.transform(X_train)
 # transform test input data
 X_test_fs = fs.transform(X_test)
 return X_train_fs, X_test_fs, fs

# feature selection
X_train_fs, X_test_fs, fs = select_features(train[["Age", "Gender", "PerfEval", "Education", "Seniority", "JobTitle", "Dept"]], train["BasePay"], test[["Age", "Gender", "Per
# what are scores for the features
for i in range(len(fs.scores_)):
 print('Feature %d: %f' % (i, fs.scores_[i]))
# plot the scores
pyplot.bar([i for i in range(len(fs.scores_))], fs.scores_)
pyplot.show()
```

```
Feature 0: 370.368230
Feature 1: 18.045034
Feature 2: 2.865274
Feature 3: 21.372117
Feature 4: 280.806801
Feature 5: 5.323906
Feature 6: 6.829043
```



Linear Regression Predictive Model

Dataset 2:

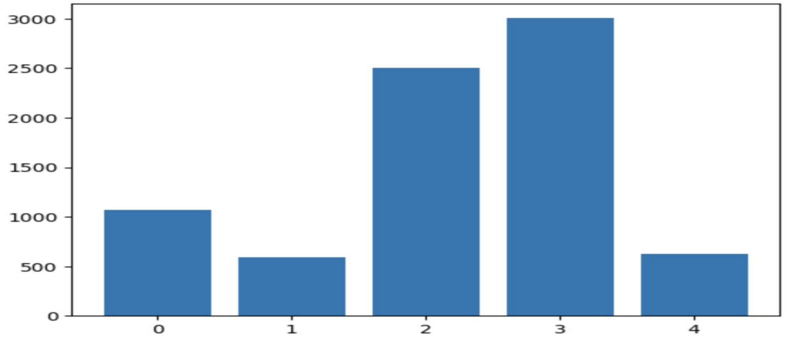
```python
train, test = train_test_split(newgender_df2, test_size=0.2, random_state=1)
model = LinearRegression()
model.fit(
    X=train[["sex", "age", "annhrs","degree", "explevel"]],
    y=train["annincome"]
)
predicted = model.predict(
    X=test[["sex", "age", "annhrs","degree", "explevel"]]
)
expected = test["annincome"]
```

	sex	age	annhrs	annincome	degree	explevel	Predicted annincome Linear
0	1	34	1600	10000.0	1.0	12.0	46781.258381
1	1	32	520	9095.0	0.0	14.0	5770.474764
2	1	64	2550	45200.0	0.0	39.0	57839.742911
3	1	50	3072	25000.0	0.0	30.0	58084.934217
4	1	26	2100	24500.0	0.0	8.0	29044.348813

```python
# feature selection
def select_features(X_train, y_train, X_test):
 # configure to select all features
 fs = SelectKBest(score_func=f_regression, k='all')
 # learn relationship from training data
 fs.fit(X_train, y_train)
 # transform train input data
 X_train_fs = fs.transform(X_train)
 # transform test input data
 X_test_fs = fs.transform(X_test)
 return X_train_fs, X_test_fs, fs

# feature selection
X_train_fs, X_test_fs, fs = select_features(train[["sex", "age", "annhrs","degree", "explevel"]], train["annincome"], test[["sex", "age", "annhrs","degree", "explevel"]])
# what are scores for the features
for i in range(len(fs.scores_)):
 print('Feature %d: %f' % (i, fs.scores_[i]))
# plot the scores
pyplot.bar([i for i in range(len(fs.scores_))], fs.scores_)
pyplot.show()
```

```
Feature 0: 1072.537640
Feature 1: 591.129558
Feature 2: 2504.987713
Feature 3: 3003.904158
Feature 4: 626.622170
```



KNN

Dataset 1:

```
x_data = df[["Age", "Gender", "PerfEval", "Education", "Seniority", "JobTitle", "Dept"]]
y_data = df["BasePay"]
X_train, X_test, y_train, y_test = train_test_split(x_data, y_data,test_size=0.2, random_state = 1)
knn_model = KNeighborsRegressor()
knn_model.fit(X_train, y_train)
predicted = knn_model.predict(X_train)
```

	JobTitle	Gender	Age	PerfEval	Education	Dept	Seniority	BasePay	Bonus	Predicted BasePay Linear	Predicted BasePay KNN
0	0	1	18	5	1	0	2	42363	9938	48713.255227	62736.8
1	1	0	21	5	1	1	5	108476	11128	91191.043229	95600.6
2	2	1	19	4	3	2	5	90208	9268	87867.125095	87303.6
3	1	0	20	5	2	3	4	108080	10154	85603.672625	100543.8
4	0	0	26	5	2	4	5	99464	9319	101417.038589	93219.6

KNN

Dataset 2:

```
x_data = newgender_df2[["sex", "age", "annhrs","degree", "explevel"]]
y_data = newgender_df2["annincome"]
X_train, X_test, y_train, y_test = train_test_split(x_data, y_data,test_size=0.2, random_state = 1)
knn_model = KNeighborsRegressor()
knn_model.fit(X_train, y_train)
predicted = knn_model.predict(X_train)
```

	sex	age	annhrs	annincome	degree	explevel	Predicted annincome Linear	Predicted annincome KNN
0	1	34	1600	10000.0	1.0	12.0	46781.258381	20960.0
1	1	32	520	9095.0	0.0	14.0	5770.474764	5095.6
2	1	64	2550	45200.0	0.0	39.0	57839.742911	76172.4
3	1	50	3072	25000.0	0.0	30.0	58084.934217	44702.4
4	1	26	2100	24500.0	0.0	8.0	29044.348813	22500.0

Bias

- Analyzed whether bias was present or not on the KNN model using Aequitas toolkit
- Cleaned data (grouped ages into age_group and experience level into experience group)
- Added is_rich column for when the income is greater than median income

	sex	age	annhrs	annincome	degree	explevel	age_group	experience_group	is_rich
0	1	34	1600	10000.0	1.0	12.0	30-40	10-15	0
1	1	32	520	9095.0	0.0	14.0	30-40	10-15	0
2	1	64	2550	45200.0	0.0	39.0	50+	25+	1
3	1	50	3072	25000.0	0.0	30.0	40-50	25+	0
4	1	26	2100	24500.0	0.0	8.0	20-30	5-10	0

Bias

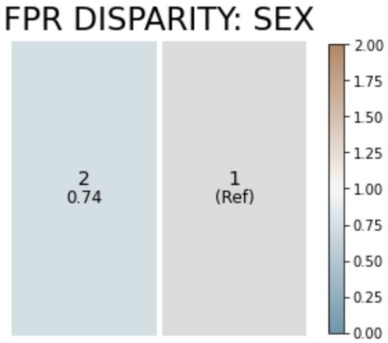
- Preprocessed data and made sure all categorical columns are of type “string”
- Created score and label_value

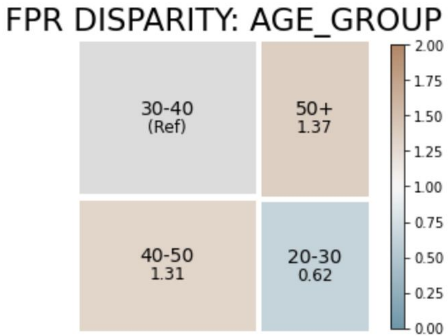
	sex	age	annhrs	degree	explevel	age_group	experience_group	score	label_value
30558	2	35	2350	0.0	15.0	30-40	15-20	0	1
20275	2	61	392	1.0	37.0	50+	25+	0	0
27706	2	43	399	1.0	15.0	40-50	15-20	0	0
195	1	53	1944	0.0	29.0	50+	25+	1	1
16981	2	31	2040	0.0	12.0	30-40	10-15	0	0

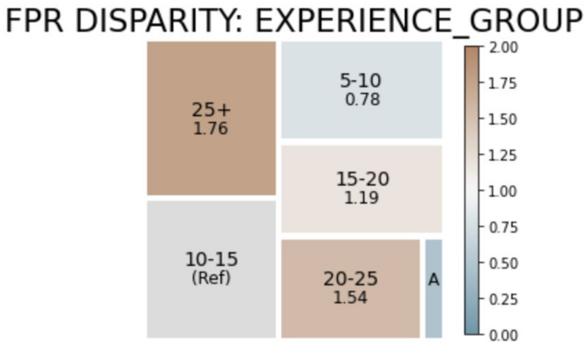
- To find bias, we choose a reference group

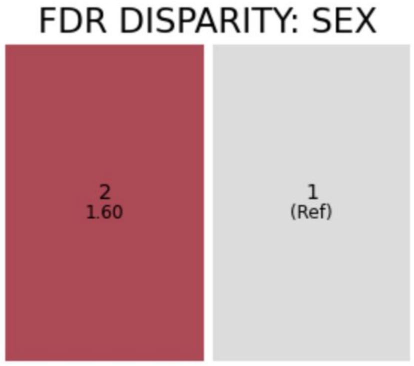
```
ref_groups_dict={'sex':'1', 'degree':'1.0', 'age_group':'30-40', 'experience_group': '10-15'}
```

Bias and Fairness









Conclusion

- There is bias present but it is more present in age and experience than sex
- This suggests that there is a disparity in pay between males and females, but it is not as large as anticipated
- Looking at the graphs, the FDR disparity shows that sex is unfair, while age and experience are generally fair