Young Adult Migration in the United States

Alaina Holland, Joselyn Molina Lagunas,
Jordyn Irwin, & Brittany Fisher

Table of contents

01 Background / Objective

02 Data

03 Models

04 Conclusions

01

Background

Background

- The importance of migration data:
  - Demographic trends
  - Labor market dynamics
  - Policy formulation and implementation
  - Global development

Target audience

- **U.S. Department of Labor & U.S. Department of Education**
- Benefits for these departments :
  - Educational planning and resource allocation
  - Identifies educational needs
  - Improves workforce development
  - Improves policy development

Objective

Problem Statement:

Predict migration patterns between origin and destination commuting zones based on race, parental income, and origin

Research Questions

1) Can parental income level and/or race predict whether or not a person will move?
2) Can parental income level and/or race predict how far a person might move?
3) Are there discernible trends based on race and parental income levels?

02

Data

Data

- Data was compiled from federal tax data from 1994, 1995, 1998-2018 linked to the 2000 and 2010 decennial censuses, 2005-2018 American Community Survey data, and Department of Housing and Urban Development address information.
- Children born 1984-1992 measuring their childhood locations at age 16 and young adult locations at age 26.
- We are focusing only on observations with an origin on the West Coast (Washington, Oregon, and California).

Count Data

	Unnamed: 0	o_cz	o_cz_name	o_state_name	d_cz	d_cz_name	d_state_name	n	n_tot_o	n_tot_d	pool	pr_d_o	pr_o_d
797036	13708961	39400	Seattle	Washington	38601	Spokane	Washington	7	2355	88	AsianQ1	0.002972	0.079546
797989	13709914	39400	Seattle	Washington	38601	Spokane	Washington	6	3569	117	AsianQ2	0.001681	0.051282
798248	13710173	39400	Seattle	Washington	38601	Spokane	Washington	9	4161	170	AsianQ3	0.002163	0.052941
798996	13710921	39400	Seattle	Washington	38601	Spokane	Washington	12	5209	152	AsianQ4	0.002304	0.078947
799614	13711539	39400	Seattle	Washington	38601	Spokane	Washington	12	5067	119	AsianQ5	0.002368	0.100840
800827	13712752	39400	Seattle	Washington	38601	Spokane	Washington	23	5444	391	BlackQ1	0.004225	0.058824
801680	13713605	39400	Seattle	Washington	38601	Spokane	Washington	34	5738	314	BlackQ2	0.005925	0.108280
801941	13713866	39400	Seattle	Washington	38601	Spokane	Washington	15	3403	227	BlackQ3	0.004408	0.066079
802942	13714867	39400	Seattle	Washington	38601	Spokane	Washington	10	2716	144	BlackQ4	0.003682	0.069444
803864	13715789	39400	Seattle	Washington	38601	Spokane	Washington	8	1790	86	BlackQ5	0.004469	0.093023

How Distance was calculated?

- Data retrieved from SimpleMaps which compiles information from the U.S. Postal Service™, U.S. Census Bureau, National Weather Service, American Community Survey, and the IRS.
- Combined on the city and state to gather the latitude and longitude.

Distance Data

```
origin = ds[(["o_cityState", "o_lat", "o_lng"])]
```

```
origin.head()
```

	o_cityState	o_lat	o_lng
0	Ontario, Oregon	44.1109	-117.0738
1	Ontario, Oregon	44.1109	-117.0738
2	Ontario, Oregon	44.1109	-117.0738
3	Ontario, Oregon	44.1109	-117.0738
4	Ontario, Oregon	44.1109	-117.0738

```
destination = ds[(["d_cityState", "d_lat", "d_lng"])]
```

```
destination.head()
```

	d_cityState	d_lat	d_lng
0	North Wilkesboro, North Carolina	36.1665	-81.0791
1	Roanoke Rapids, North Carolina	36.4305	-77.7183
2	Monroe, Louisiana	32.5392	-92.1069
3	Hattiesburg, Mississippi	31.2350	-89.2691
4	Austin, Texas	30.2702	-97.7425

Distance data

```python
origin[['lat_radians_o','long_radians_o']] = (
    np.radians(origin.loc[:,['o_lat','o_lng']])
)
destination[['lat_radians_d','long_radians_d']] = (
    np.radians(destination.loc[:,['d_lat','d_lng']])
)
```

```python
origin=origin.drop_duplicates()
origin.head()
```

	o_cityState	o_lat	o_lng	lat_radians_o	long_radians_o
0	Ontario, Oregon	44.1109	-117.0738	0.769880	-2.043323
18525	Klamath Falls, California	41.5863	-124.0586	0.725818	-2.165231
37050	Burns, Oregon	43.5880	-118.8581	0.760754	-2.074465
55575	Lakeview, Oregon	42.2983	-120.3917	0.738245	-2.101232
74100	Redding, California	40.5773	-122.4544	0.708207	-2.137232

```python
destination=destination.drop_duplicates()
destination.head(10)
```

	d_cityState	d_lat	d_lng	lat_radians_d	long_radians_d
0	North Wilkesboro, North Carolina	36.1665	-81.0791	0.631225	-1.415097
1	Roanoke Rapids, North Carolina	36.4305	-77.7183	0.635832	-1.356440
2	Monroe, Louisiana	32.5392	-92.1069	0.567916	-1.607569
3	Hattiesburg, Mississippi	31.2350	-89.2691	0.545154	-1.558040
4	Austin, Texas	30.2702	-97.7425	0.528315	-1.705928

```python
from sklearn.neighbors import DistanceMetric
dist = DistanceMetric.get_metric('haversine')

dist_matrix = (sklearn.metrics.pairwise.haversine_distances
    (origin[['lat_radians_o','long_radians_o']],
    destination[['lat_radians_d','long_radians_d']])*3959
)
```

Distance data

```
ds_dist_unpv.head()
```

	o_cityState	d_cityState	distance
0	Ontario, Oregon	North Wilkesboro, North Carolina	1961.671028
1	Klamath Falls, California	North Wilkesboro, North Carolina	2317.821008
2	Burns, Oregon	North Wilkesboro, North Carolina	2048.864779
3	Lakeview, Oregon	North Wilkesboro, North Carolina	2126.734491
4	Redding, California	North Wilkesboro, North Carolina	2241.271000

```
conv_fac = 0.621371
ds_dist_unpv["miles"] = ds_dist_unpv.distance /conv_fac
```

```
ds_dist_unpv.head()
```

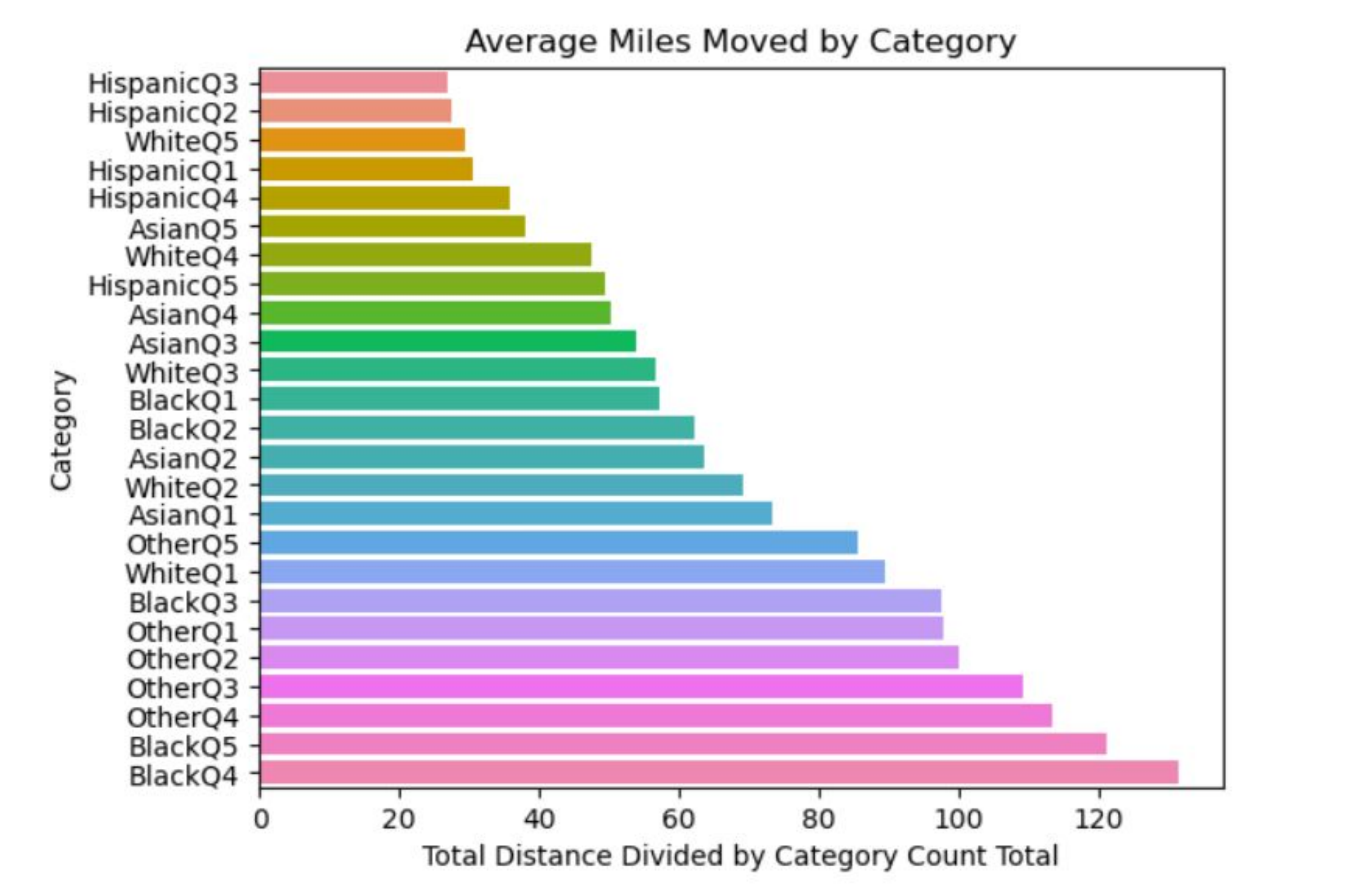
	o_cityState	d_cityState	distance	miles
0	Ontario, Oregon	North Wilkesboro, North Carolina	1961.671028	3157.004476
1	Klamath Falls, California	North Wilkesboro, North Carolina	2317.821008	3730.172487
2	Burns, Oregon	North Wilkesboro, North Carolina	2048.864779	3297.329259
3	Lakeview, Oregon	North Wilkesboro, North Carolina	2126.734491	3422.648452
4	Redding, California	North Wilkesboro, North Carolina	2241.271000	3606.977153

Transformations

- Narrowed to and origin only on the West Coast– California, Oregon, and Washington
- Count data expanded into individual record rows
- Joined with the latitude and longitude of both the origin and the destination
- Categorical data re-coded
- Migration column created

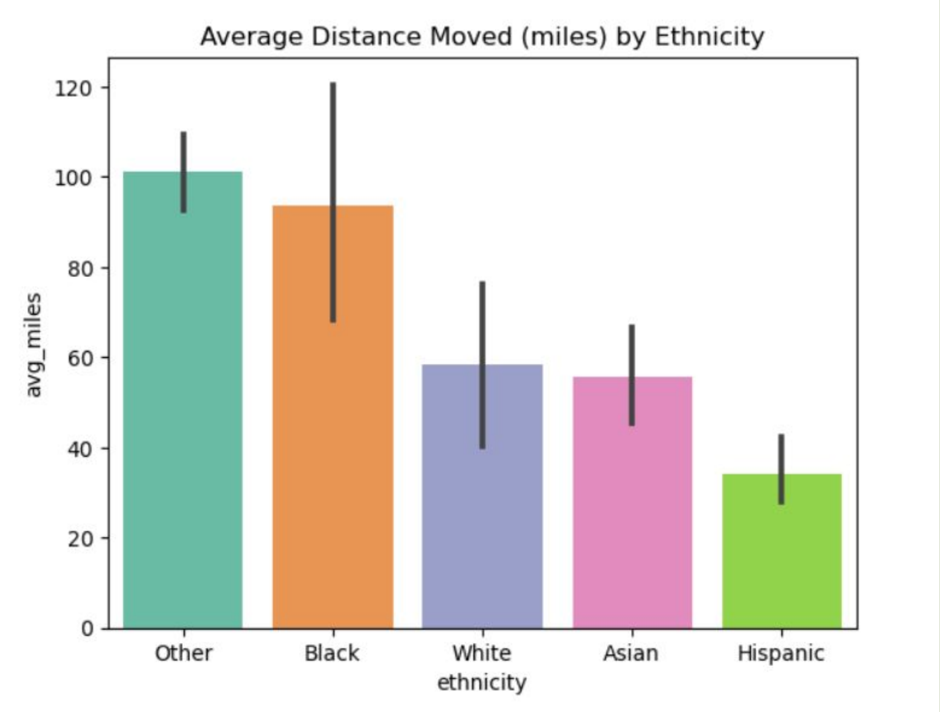
	n	pool	ethnicity	income	migrated	d_cityState	d_lat	d_lng	o_cityState	o_lat	o_lng	o_d	dist_miles
0	0	AsianQ1	1	1	1	North Wilkesboro, North Carolina	36.1665	-81.0791	Ontario, Oregon	44.1109	-117.0738	Ontario, Oregon-North Wilkesboro, North Carolina	3157.004476
1	0	AsianQ1	1	1	1	Roanoke Rapids, North Carolina	36.4305	-77.7183	Ontario, Oregon	44.1109	-117.0738	Ontario, Oregon-Roanoke Rapids, North Carolina	3411.728192
2	0	AsianQ1	1	1	1	Monroe, Louisiana	32.5392	-92.1069	Ontario, Oregon	44.1109	-117.0738	Ontario, Oregon-Monroe, Louisiana	2514.796149
3	0	AsianQ1	1	1	1	Hattiesburg, Mississippi	31.2350	-89.2691	Ontario, Oregon	44.1109	-117.0738	Ontario, Oregon-Hattiesburg, Mississippi	2815.030003
4	0	AsianQ1	1	1	1	Austin, Texas	30.2702	-97.7425	Ontario, Oregon	44.1109	-117.0738	Ontario, Oregon-Austin, Texas	2291.772825

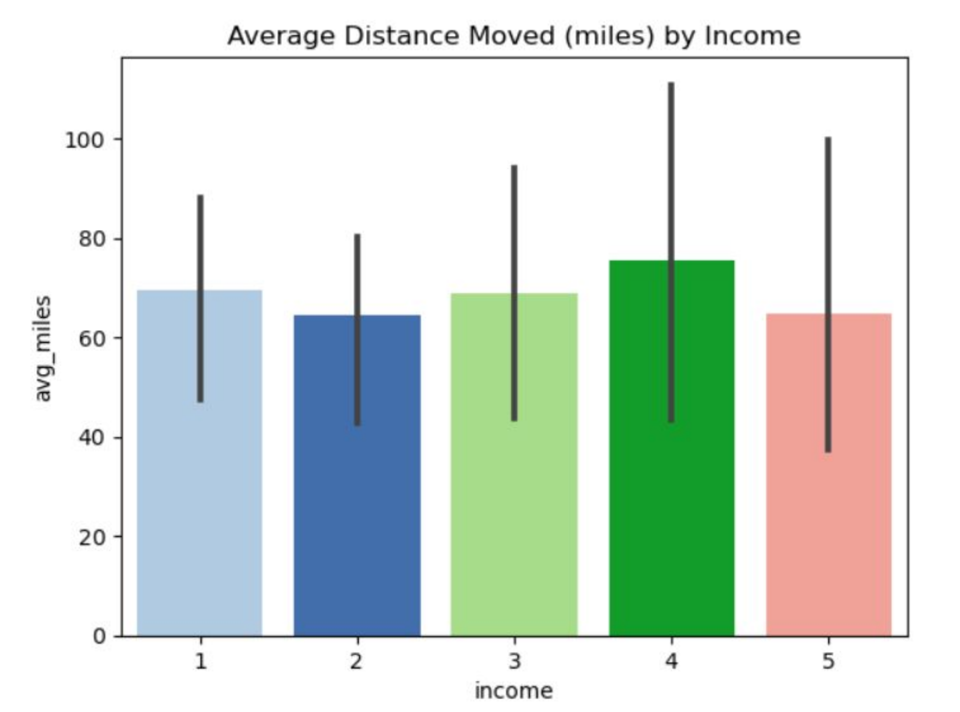
EDA



Average Miles Moved by Category
HispanicQ3
HispanicQ2
WhiteQ5
HispanicQ1
HispanicQ4
AsianQ5
WhiteQ4
HispanicQ5
AsianQ4
AsianQ3
WhiteQ3
BlackQ1
BlackQ2
AsianQ2
WhiteQ2
AsianQ1
OtherQ5
WhiteQ1
BlackQ3
OtherQ1
OtherQ2
OtherQ3
OtherQ4
BlackQ5
BlackQ4
Category
0
20
40
60
80
100
120
Total Distance Divided by Category Count Total

Differences in Average Distance Moved by Ethnicity and Income





03

Models

RQ 1:
Can we predict if a young person will move based on their ethnicity, parental income, and origin city?

Random Forest Classifier

Details:

- Data is very unbalanced: 1,261,736 individuals who did migrate and 3,362,219 who did not.
- To combat this, the train test split includes random_state=1 and stratify=y.
- Feature variables: pooled ethnicity and parental income, ethnicity, parental income, their origin city
- Response variable: their migration status
- Again, to account for the unbalanced data set, a weight class of balanced was set.

```
X_train1, X_test1, y_train1, y_test1 = train_test_split(x, y, random_state=1, stratify=y)
print (X_train1.shape, y_train1.shape)
print (X_test1.shape, y_test1.shape)
```

```
(3467966, 4) (3467966,)
(1155989, 4) (1155989,)
```

Random Forest Classifier

Results:

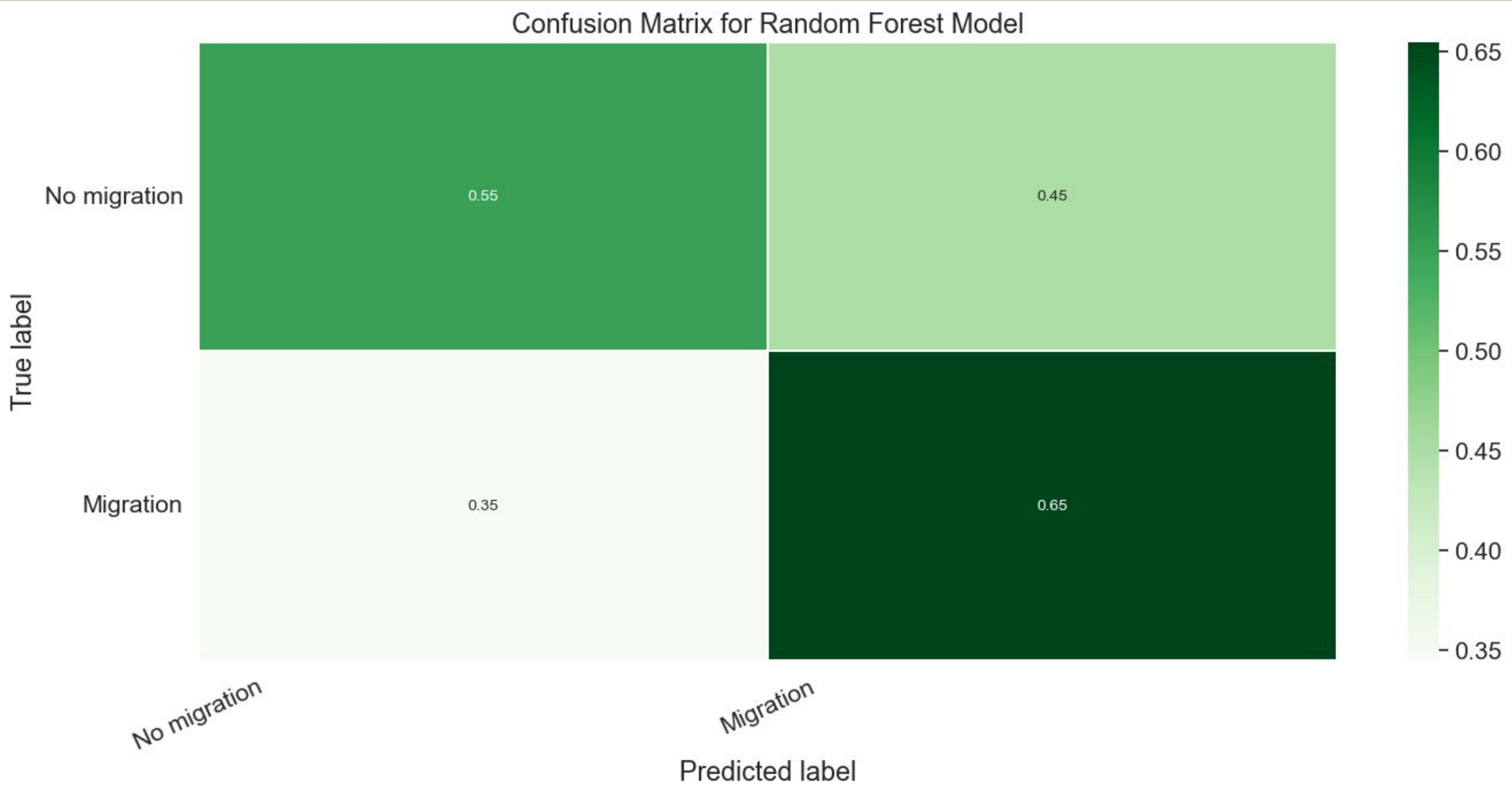
- Meh!
- Accuracy score: 0.5784328397588558
- Not terribly indicative for this algorithm
- Precision for non-migration is great, but can only predict the migrating youth 35% of the time.
- Recall is better for migrating youth and both are in the ‘meh’ realm.
- F-1 score falls into average overall.

F1 score	Interpretation
> 0.9	Very good
0.8 - 0.9	Good
0.5 - 0.8	OK
< 0.5	Not good

```
print(classification_report(y_test1, preds))
```

```
              precision    recall  f1-score   support

           0       0.81      0.55      0.65    840555
           1       0.35      0.65      0.46    315434

    accuracy                           0.58   1155989
   macro avg       0.58      0.60      0.56   1155989
weighted avg       0.68      0.58      0.60   1155989
```

Random Forest Classifier



K-Nearest Neighbors

Details:

- Due to extremely slow processing, the dataset needed to be smaller.
- To accomplish this, we took a stratified sample of 20,000 per pool from 4.6 million rows, which gives us 500,000 total rows in the sample dataset.
- Algorithm set to 'kd_tree' to optimize— even a sample still took quite a long time.
- K set to 3– choice confirmed with further analyzation.

```
knn_clf = KNeighborsClassifier(weights='distance', n_neighbors=3, algorithm ='kd_tree')
knn_clf.fit(X_train, y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier(algorithm='kd_tree', n_neighbors=3, weights='distance')
```

K-Nearest Neighbors

Results:

- Not fantastic!
- Accuracy score: 0.683584
- Again, precision, recall, and F-1 scores are worse for the migrating youth.
- Overall average scores fall into the middle of the road.

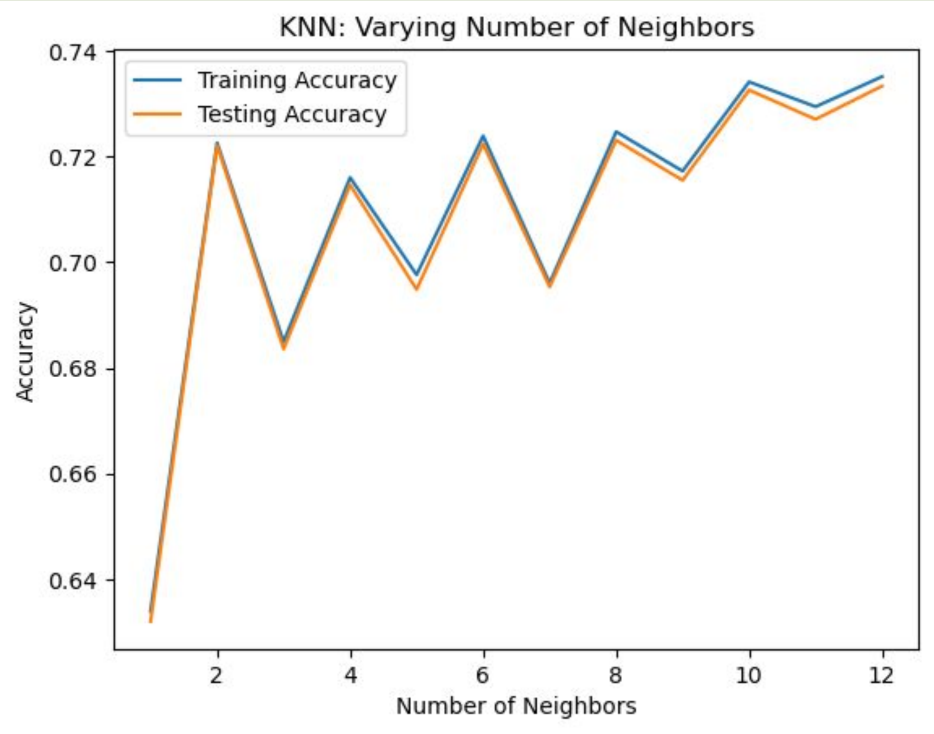
F1 score	Interpretation
> 0.9	Very good
0.8 - 0.9	Good
0.5 - 0.8	OK
< 0.5	Not good

```
print(classification_report(y_test, y_knn_pred))
```

```
              precision    recall  f1-score   support

           0       0.75      0.86      0.80     92200
           1       0.33      0.20      0.24     32800

    accuracy                           0.68    125000
   macro avg       0.54      0.53      0.52    125000
weighted avg       0.64      0.68      0.65    125000
```

K-Nearest Neighbors Continued:

- Is three the correct amount of clusters?
- We used a function to loop through 12 cluster options:
- Testing accuracies for each number of clusters:
  - 1: 0.631992, 2: 0.722112, 3: 0.683584, 4: 0.714632, 5: 0.694856, 6: 0.722416, 7: 0.695352, 8: 0.72312, 9: 0.715536, 10: 0.732624, 11: 0.72708, 12: 0.733384
- Though accuracy improves with even clusters, precision and recall tank even further for migrating individuals
  - Odd numbers also recommended in case of a tie



Random Forest Classifier

Continued:

- To align with the sampled data used in the KNN classifier, we re-ran the Random Forest Classifier with that data.
- Extremely similar to the entire dataset.

F1 score	Interpretation
> 0.9	Very good
0.8 - 0.9	Good
0.5 - 0.8	OK
< 0.5	Not good

```
print(classification_report(y_test, preds2))
```

```
              precision    recall  f1-score   support

           0       0.81      0.56      0.66     92200
           1       0.34      0.63      0.44     32800

    accuracy                           0.58    125000
   macro avg       0.57      0.59      0.55    125000
weighted avg       0.69      0.58      0.61    125000
```

Categorical Naive Bayes

Details:

- Naive Bayes are a family of linear "probabilistic classifiers".
- A "simple" classifier.
- Assumes that the features are conditionally independent.

```
cnb = CategoricalNB()
cnb.fit(X_train, y_train)
```

```
▾ CategoricalNB
CategoricalNB()
```

```
y_predict = cnb.predict(X_test)
```

Categorical Naive Bayes

Results:

- Misleading!
- Again, the model is having a difficult time with the migrating youth.
- The overall accuracy is an improvement on previous models, but the avg for recall and precision are very similar to previous models.

```
accuracy_score(y_test, y_predict)
```

0.7256

```
print(classification_report(y_test, y_predict))
```

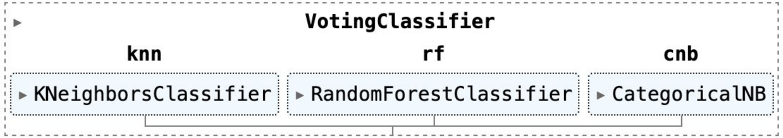
	precision	recall	f1-score	support
0	0.75	0.94	0.83	92200
1	0.43	0.13	0.20	32800
accuracy			0.73	125000
macro avg	0.59	0.53	0.52	125000
weighted avg	0.67	0.73	0.67	125000

Ensemble Learning

Voting classifier:

- A technique to enhance models and their ability to predict by merging predictions from multiple models.
- Gains experience by training on a collection of several models and predicting a class based on the class with the highest likelihood of becoming the output.
- Two types: hard voting and soft voting

```
voting_clf.fit(X_train, y_train)
```

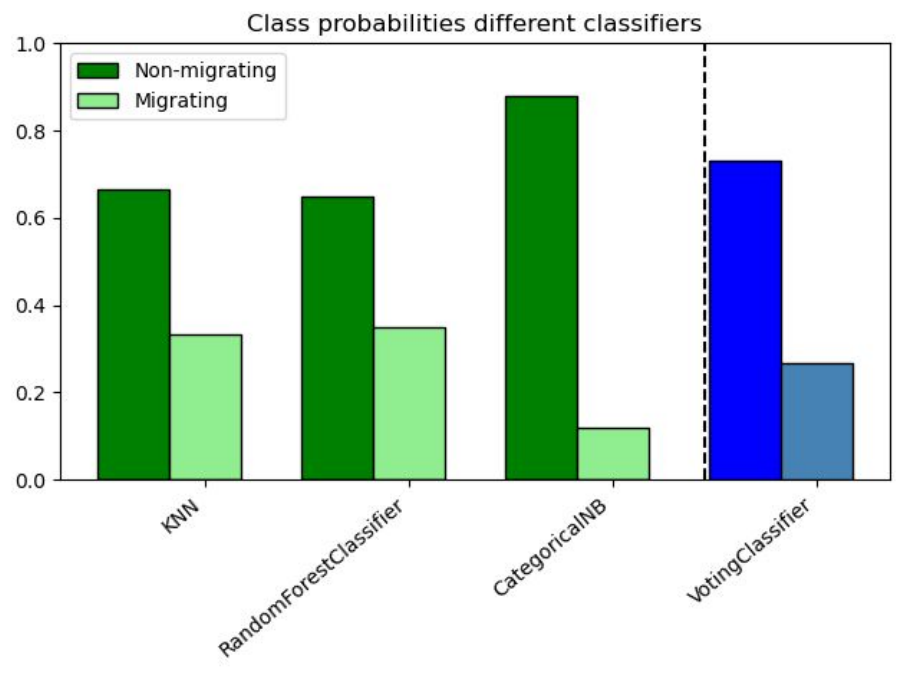



```
for clf3 in (knn_clf, clf2, cnb, voting_clf):
    clf3.fit(X_train, y_train)
    y_pred = clf3.predict(X_test)
    print(clf3.__class__.__name__, accuracy_score(y_test, y_pred))
```

```
KNeighborsClassifier 0.634504
RandomForestClassifier 0.589464
CategoricalNB 0.728208
VotingClassifier 0.680464
```

Ensemble Learning

Results:



- The KNN and Random Forest Classifier are extremely similar.
- Using the Voting Classifier operates under the bias-variance trade off concept.

Model Performance

KNeighborsClassifier

	precision	recall	f1-score	support
0	0.75	0.83	0.79	92129
1	0.32	0.23	0.27	32871
accuracy			0.67	125000
macro avg	0.54	0.53	0.53	125000
weighted avg	0.64	0.67	0.65	125000

RandomForestClassifier

	precision	recall	f1-score	support
0	0.80	0.60	0.69	92129
1	0.34	0.58	0.43	32871
accuracy			0.60	125000
macro avg	0.57	0.59	0.56	125000
weighted avg	0.68	0.60	0.62	125000

CategoricalNB

	precision	recall	f1-score	support
0	0.75	0.93	0.83	92129
1	0.42	0.13	0.20	32871
accuracy			0.72	125000
macro avg	0.58	0.53	0.52	125000
weighted avg	0.66	0.72	0.67	125000

VotingClassifier

	precision	recall	f1-score	support
0	0.76	0.89	0.82	92129
1	0.39	0.20	0.27	32871
accuracy			0.71	125000
macro avg	0.57	0.54	0.54	125000
weighted avg	0.66	0.71	0.67	125000

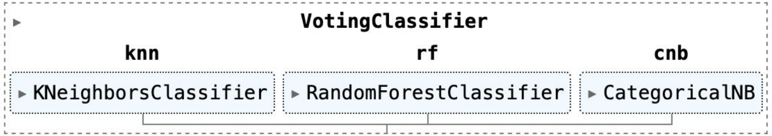
RQ 2:
Can we predict how far a young person will move based on their ethnicity, parental income, and origin city?

Random Forest Regressor

Details:

- Feature variables: pooled ethnicity and parental income, ethnicity, parental income, their origin city, and migration status.
- Response variable: distance moved in miles.
- Still using the 20k stratified sample.

```python
voting_clf.fit(X_train, y_train)
```




```python
for clf3 in (knn_clf, clf2, cnb, voting_clf):
    clf3.fit(X_train, y_train)
    y_pred = clf3.predict(X_test)
    print(clf3.__class__.__name__, accuracy_score(y_test, y_pred))
```

```
KNeighborsClassifier 0.634504
RandomForestClassifier 0.589464
CategoricalNB 0.728208
VotingClassifier 0.680464
```

Random Forest Regressor

Results:

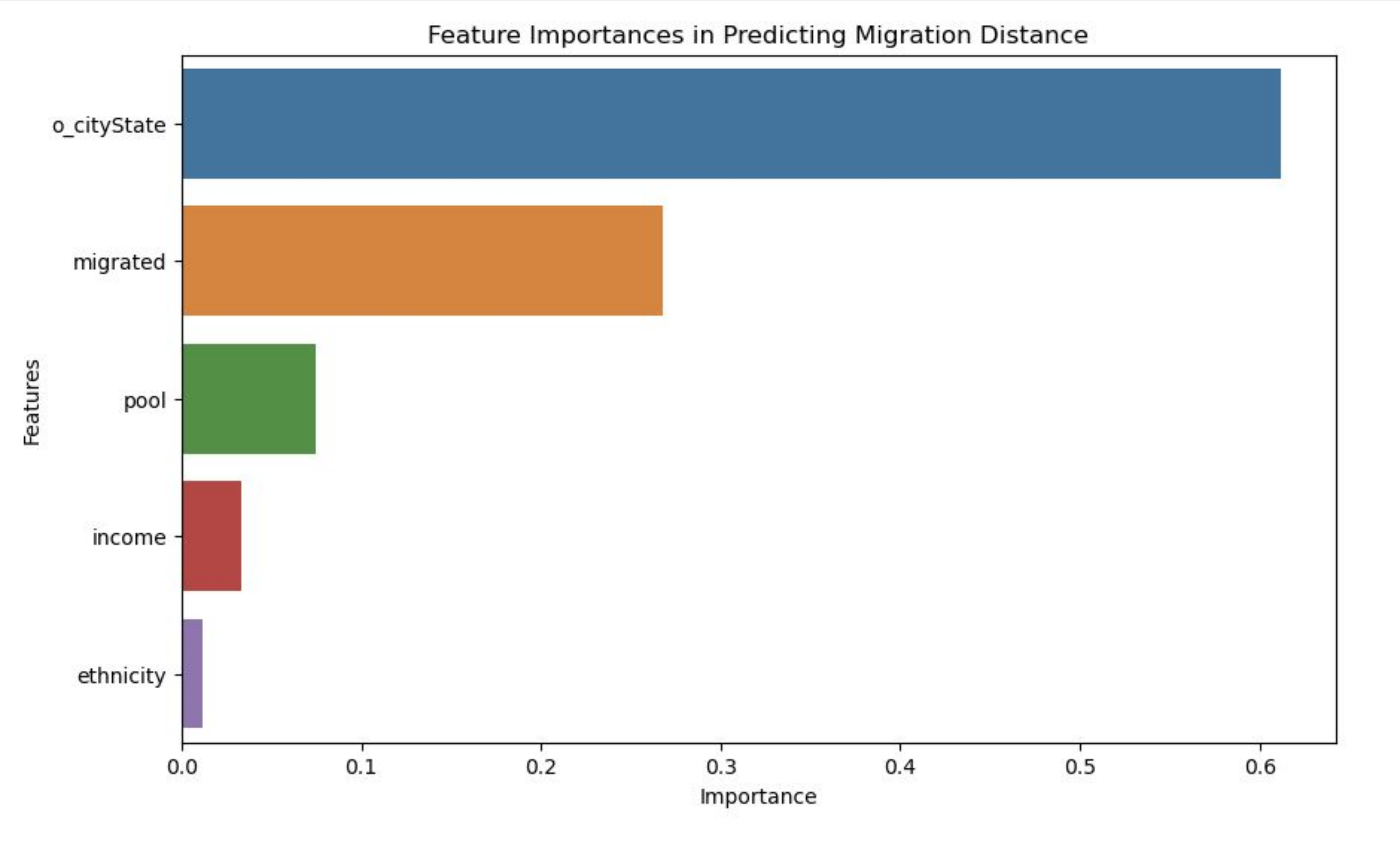
- So bad!
- We wanted a mean squared error as close to 0 as possible.
- Also, we wanted an R-squared close to 1.

```
oob_score = distCLF.oob_score_
print(f'Out-of-Bag Score: {oob_score}')
```

```
Out-of-Bag Score: 0.025392197181454135
```

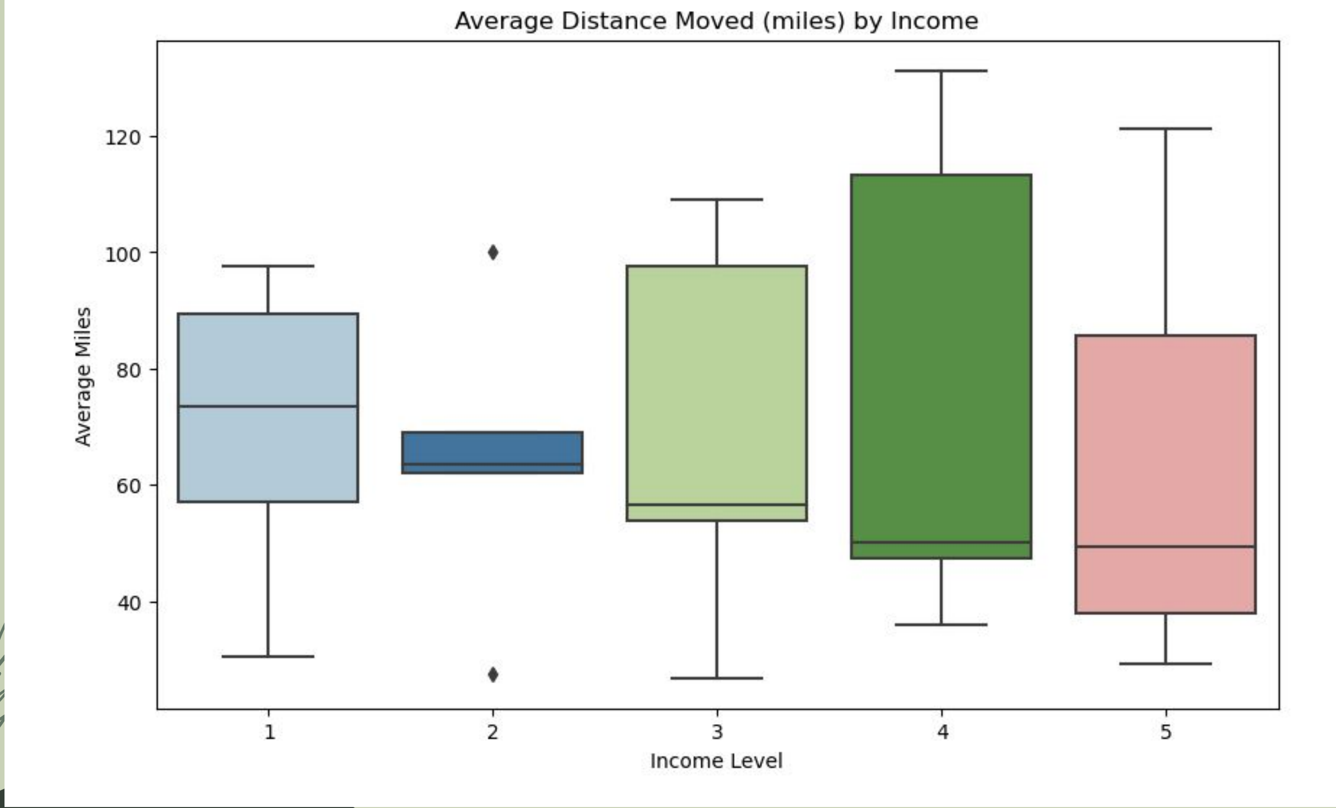
```
mse = mean_squared_error(y_test, distPred)
print(f'Mean Squared Error: {mse}')

r2 = r2_score(y_test, distPred)
print(f'R-squared: {r2}')
```

```
Mean Squared Error: 893784.4470626586
R-squared: 0.02581330523197678
```


Feature Importances in Predicting Migration Distance
o_cityState
migrated
pool
income
ethnicity
Features
0.0
0.1
0.2
0.3
0.4
0.5
0.6
Importance

RQ 3:
Are there discernible trends based on race and family income levels?

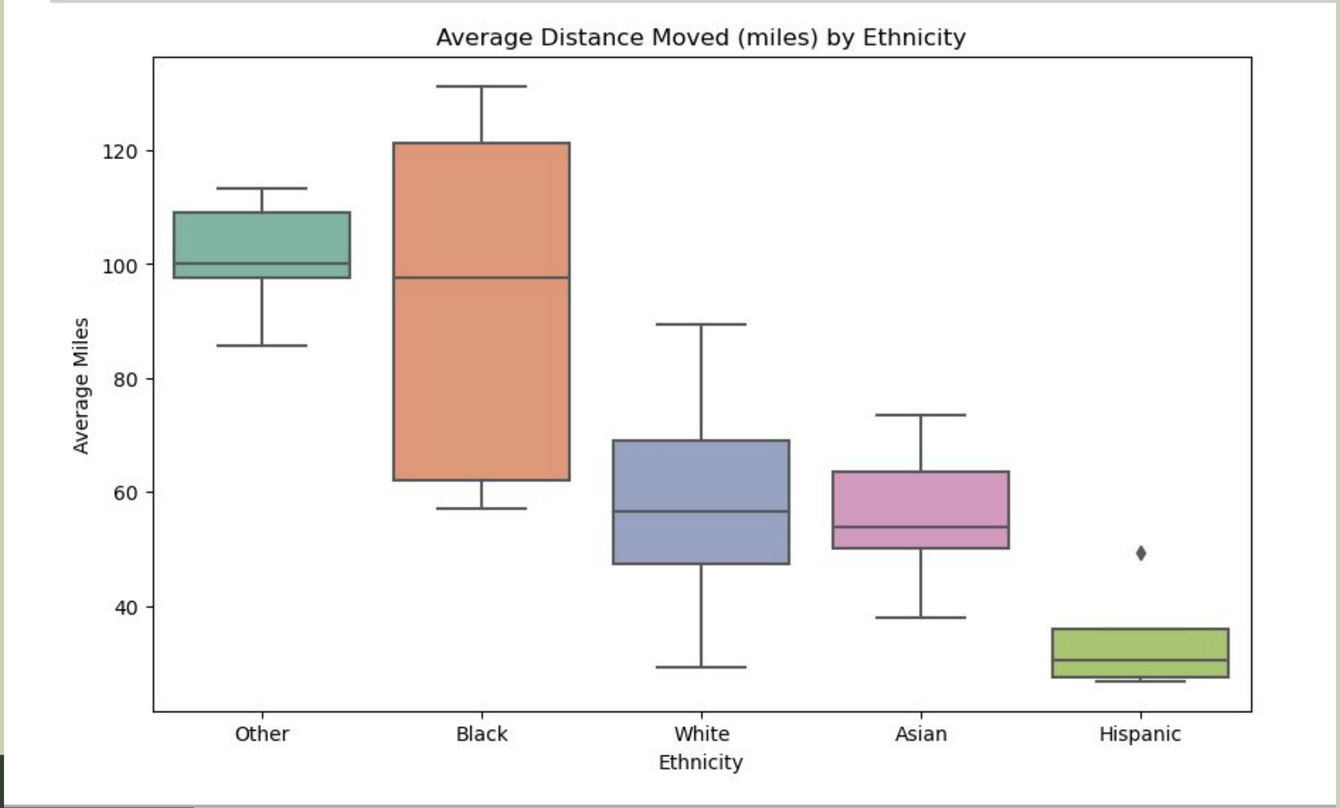
Parental Income



Note: Income Quartile is based on the ranges per city NOT U.S. average income quartiles

Quartile	Income
Q1	Lowest
Q2	Lower-Middle
Q3	Middle
Q4	Middle-High
Q5	Highest

Race



Average Distance Moved (miles) by Ethnicity
Average Miles
120
100
80
60
40
Other
Black
White
Asian
Hispanic
Ethnicity

04

Conclusions

Conclusions

Results:

- Our ability to predict migration for youth from this data is pretty mediocre.
- Combined models performed the best.
- We cannot predict how far someone moves just with this data.

Limitations:

- Count data!
- Very few variables to explore.
- Categorical variables
- Somewhat out of date.
- Limited ML experience

Next Steps

Data:

- At what age did they move?
- What are their motivations to move?
- Is it due to college? If so, do they return to their origin city?

Incorporate Additional Variables:

- Employment opportunities
- Housing prices
- Education levels
- Family ties
- Environmental factors like climate or geographic features
- Time Period of move (economic recession/boom)
- Expand to across the country

Questions:

- Are there important differences between age groups?
- Have there been changes over time?
- Did the pandemic make a significant difference to migration?

Any questions?

PS: pls don't have questions,
we are very tired.