codecademy

Machine Learning Capstone Project

John Gimlin
jdgimlin@gmail.com
09/01/2020

SLIDESMANIA.COM

Project Overview

For this capstone project, codecademy provided a dataset from **okcupid**, an online dating app that matches users based on a number of multiple choice and short-answer essay questions.

From codecademy:
"The purpose of this capstone is to practice formulating questions and implementing Machine Learning techniques to answer those questions."

SLIDESMANIA.COM

Project Tasks

1. Explore the Data
2. Visualize the Data
3. Formulate a Question
4. Augment the Data
5. Compare Classification Models
6. Compare Regression Models
7. Analyze Model Metrics
8. Conclusion

SLIDESMANIA.COM

1. Explore the Data

SLIDESMANIA.COM

1. Explore the Data

The okcupid dataset consists of:

- **59,946** rows and **31** columns
- **3** numerical columns
- **28** object data-type columns

Columns

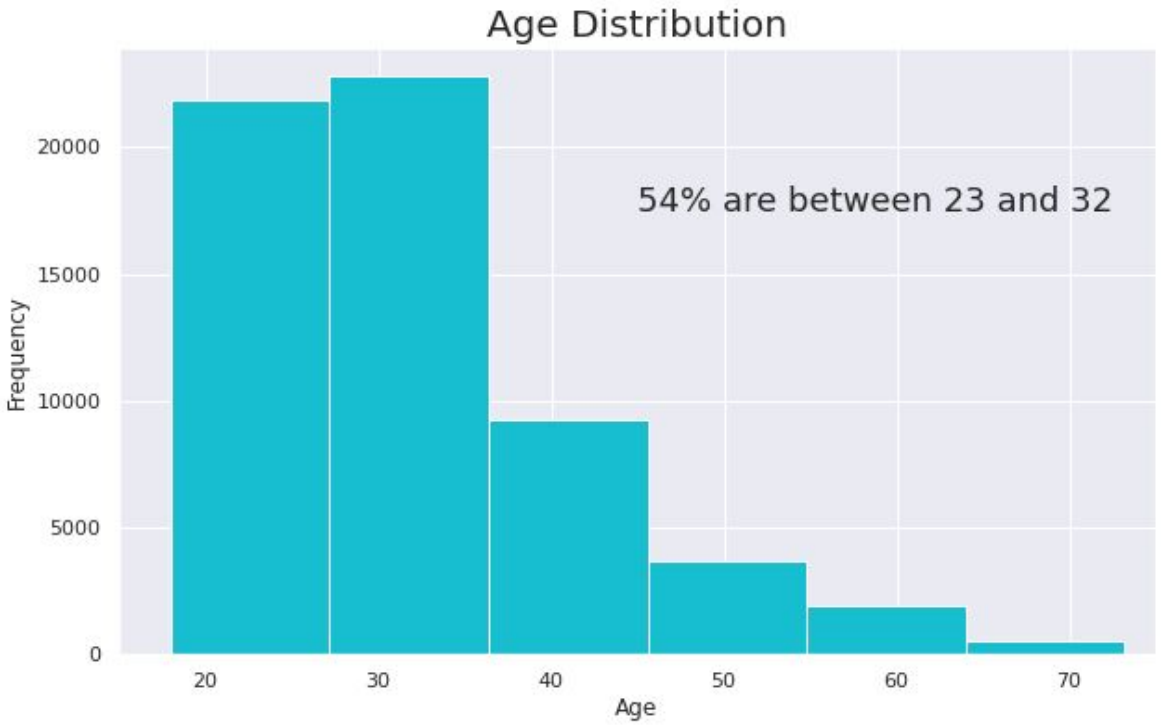
['age', 'body_type', 'diet', 'drinks', 'drugs', 'education', 'essay0', 'essay1', 'essay2', 'essay3', 'essay4', 'essay5', 'essay6', 'essay7', 'essay8', 'essay9', 'ethnicity', 'height', 'income', 'job', 'last_online', 'location', 'offspring', 'orientation', 'pets', 'religion', 'sex', 'sign', 'smokes', 'speaks', 'status']

SLIDESMANIA.COM

2. Visualize the Data

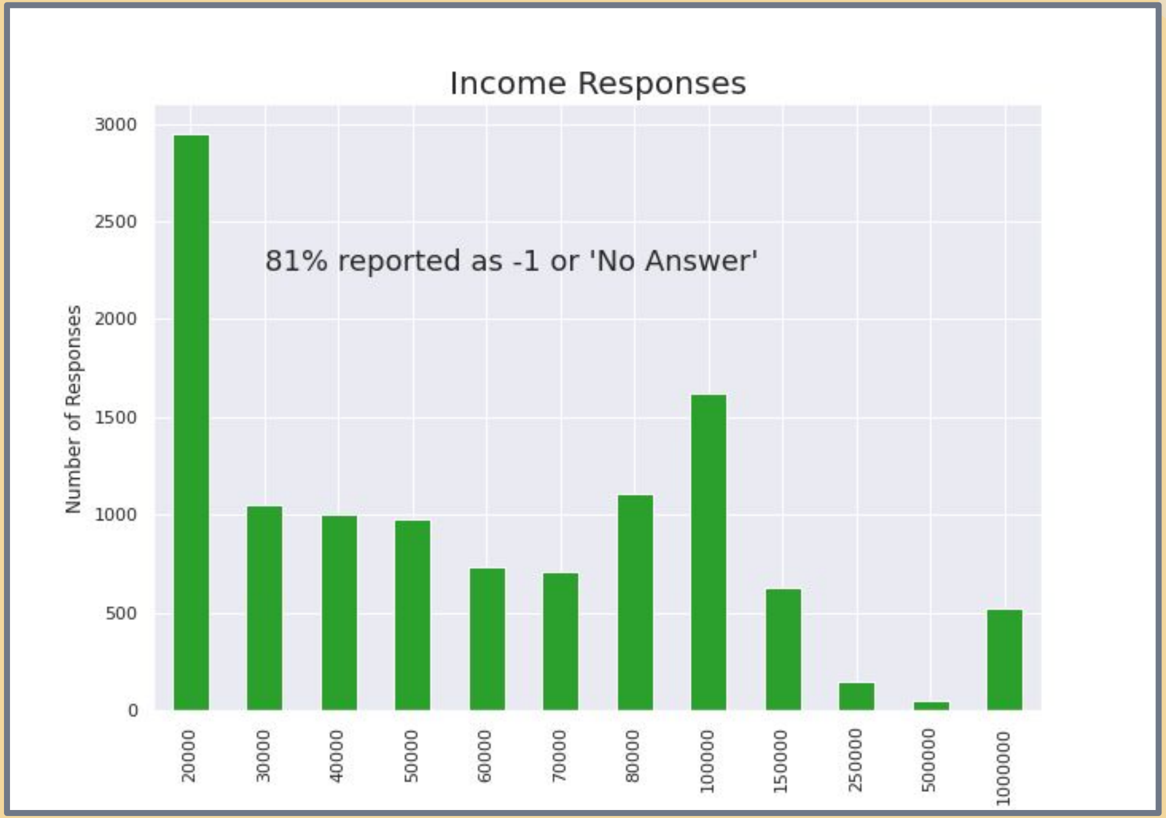
SLIDESMANIA.COM

Visualize the Data



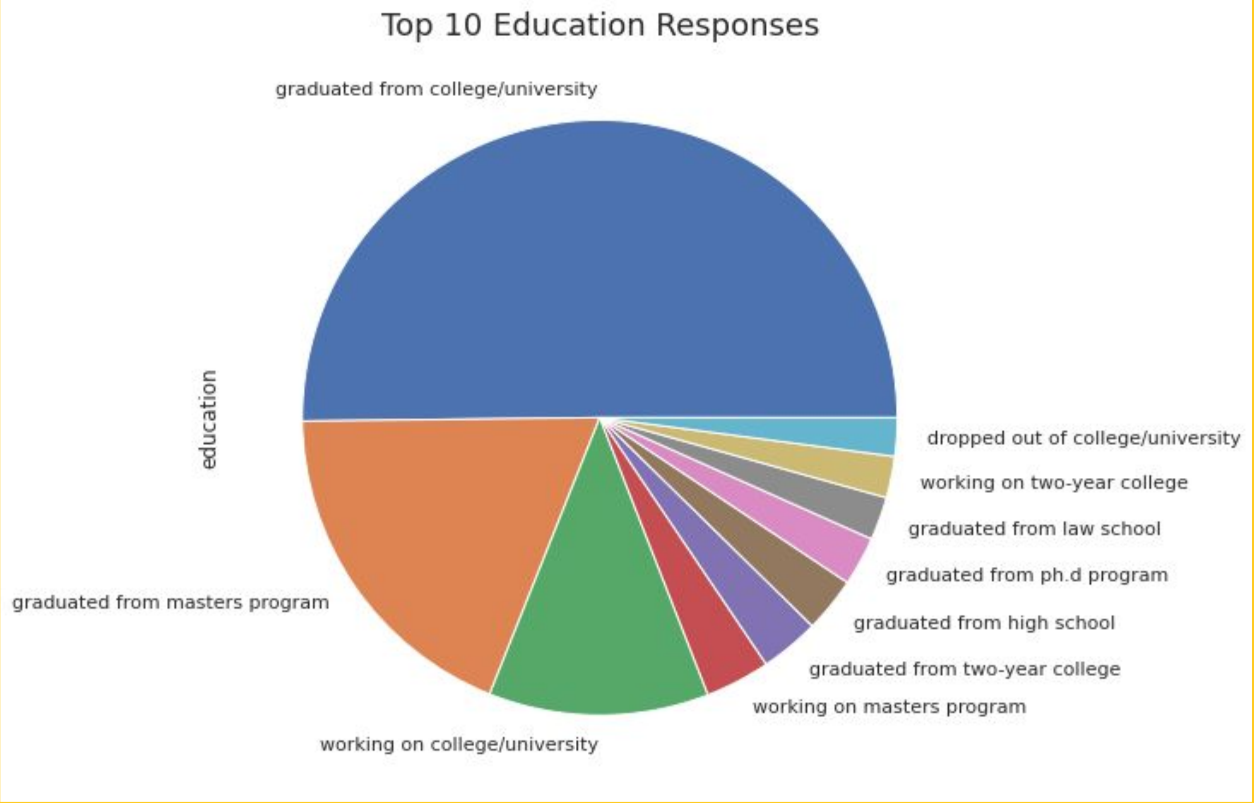
SLIDESMANIA.COM

Visualize the Data



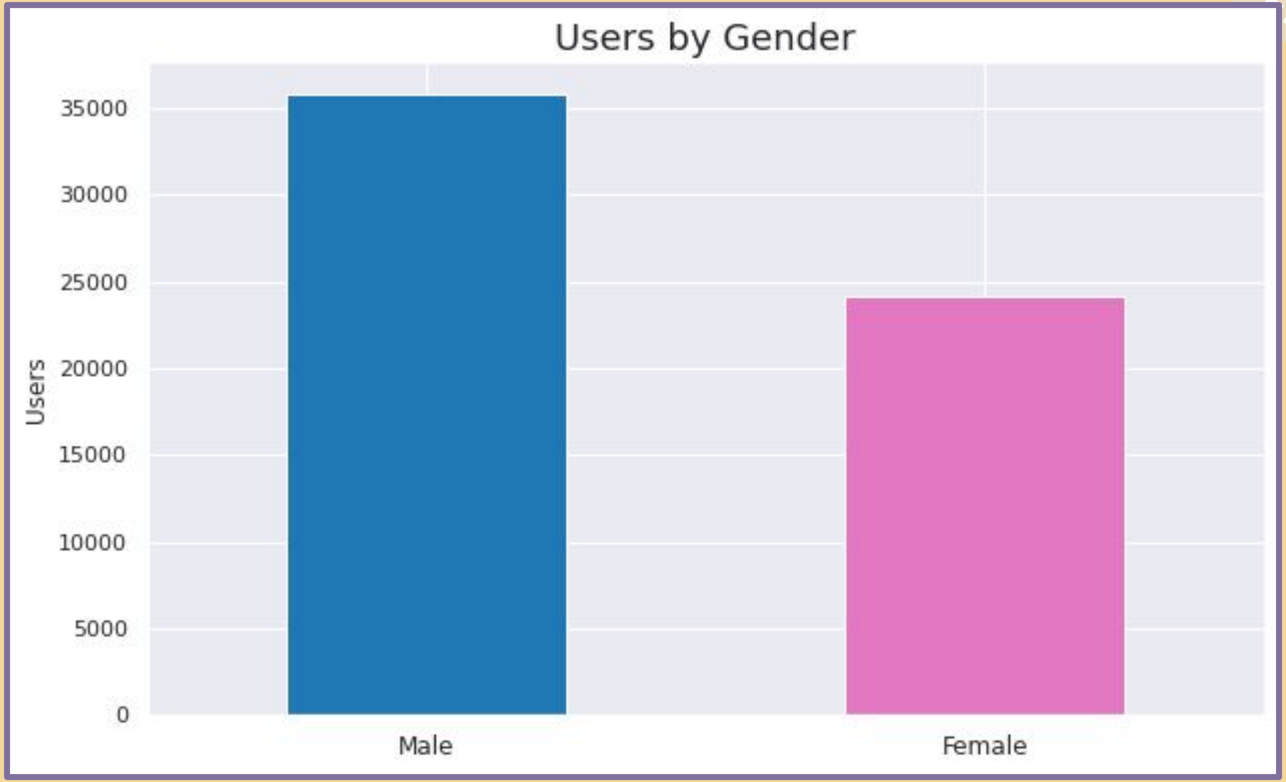
SLIDESMANIA.COM

Visualize the Data



SLIDESMANIA.COM

Visualize the Data



SLIDESMANIA.COM

3. Formulate a Question

SLIDESMANIA.COM

3. Formulate a Question

Classification Model Question: Can we predict gender based on the following features:

1. Whether a user prefers cats or dogs from the pets column
2. A user's body_type and height
3. How much a user drinks or smokes

I chose to predict gender because it's a binary, categorical response. After trying many different features, I chose the above out of general interest and possible correlation to the response category.

SLIDESMANIA.COM

3. Formulate a Question

Regression Model Question: Can we predict height based on the following features:

1. A user's diet
2. Their drug use
3. Their gender, or sex

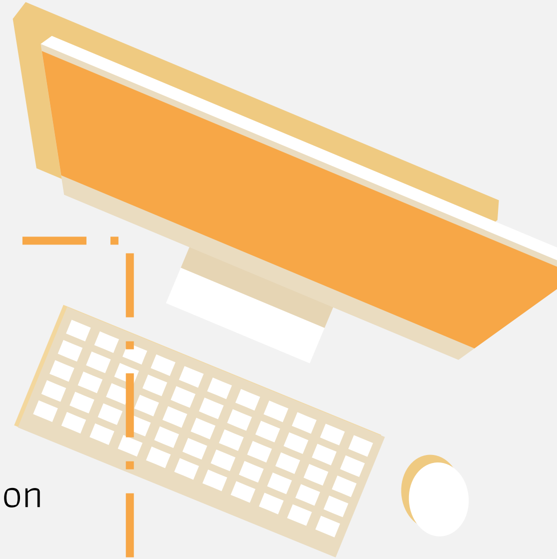
I was limited to only three possible regression response columns. I didn't choose income due to the high number of null values. That left age or height. I chose height because it had more correlation to the feature columns.

SLIDESMANIA.COM

4. Augment the Data

SLIDESMANIA.COM

4. Augment the Data



A note on classifying categorical data with numeric values:

Codecademy does not cover any preprocessing techniques or tools, such as dummy variables with pandas or scikit learn’s OneHotEncoder. As a result I mapped categorical responses into new columns using integers based solely on common sense and instinct.

This was complicated by the fact the okcupid survey contained many possible answers for essentially the same response across all of the categorical response questions.

As a project requirement the next slide will show how I chose to map some of the feature categories used in the modeling. This method was used for the classification and regression models.

SLIDESMANIA.COM

4. Augment the Data

Sample mapping of data into new columns:

```python
# Map "pets"
# Create & map feature column: pets_code

pets_mapping = {
    "likes dogs and likes cats": 1,
    "likes dogs": 0,
    "likes dogs and has cats": 2,
    "has dogs": 0,
    "has dogs and likes cats": 0,
    "likes dogs and dislikes cats": 0,
    "has dogs and has cats": 1,
    "has cats": 2,
    "likes cats": 2,
    "has dogs and dislikes cats": 0,
    "dislikes dogs and likes cats": 2,
    "dislikes dogs and dislikes cats": -1,
    "dislikes cats": -1,
    "dislikes dogs and has cats": 2,
    "dislikes dogs": -1}
df_class["pets_code"] = df_class.pets.map(pets_mapping)
```

```python
# Map "body_type"
# Create & map feature column: body_type_code

bt_mapping = {
    "average": 0,
    "fit": 1,
    "athletic": 1,
    "thin": -1,
    "curvy": 2,
    "a little extra": 2,
    "skinny": -1,
    "full figured": 2,
    "overweight": 3,
    "jacked": 1,
    "used up": -1,
    "rather not say": 0}
df_class["body_type_code"] = df_class.body_type.map(bt_mapping)
```

SLIDESMANIA.COM

4. Augment the Data

Sample mapping of data into new columns:

```
# Map "drinks"
# Create & map feature column: drinks_code

drinks_mapping = {
    "socially": 0,
    "rarely": 1,
    "often": 2,
    "not at all": -1,
    "very often": 3,
    "desperately": 3}
df_class["drinks_code"] = df_class.drinks.map(drinks_mapping)
```

```
# Map "smokes"
# Create & map feature column: smokes_code

smokes_mapping = {
    "no": 0,
    "sometimes": 1,
    "when drinking": 1,
    "yes": 2,
    "trying to quit": 2,}
df_class["smokes_code"] = df_class.smokes.map(smokes_mapping)
```

```
# Map response column y: "sex"
# Create new column: sex_code

df_class["sex_code"] = df_class.loc[:, "sex"].apply(lambda x: 1 if x == "m" else 0)
```

SLIDESMANIA.COM

4. Augment the Data (Classification Model)

New column created: sex_code

This served as the response column where "male" was mapped as 1 and "female" as 0.

```
df_class.sex_code.value_counts()

1     19900
0     14522
Name: sex_code, dtype: int64
```

New column created: pets_code

Mapped user preferences for cats, dogs, both, and neither as numerical values.

```
df_class.pets_code.value_counts()

 1     14122
 0     13945
 2      6026
-1       329
Name: pets_code, dtype: int64
```

New column created: body_type_code

Mapped user responses for thin, average, fit, etc. as numerical values.

```
df_class.body_type_code.value_counts()

 1     14888
 0      9465
 2      5520
-1      4209
 3       340
Name: body_type_code, dtype: int64
```

SLIDESMANIA.COM

4. Augment the Data (Classification Model) cont.

New column created: drinks_code

Mapped user responses for drinking alcohol into numerical values.

New column created: smokes_code

Mapped user responses for smoking as numerical values.

```
df_class.drinks_code.value_counts()

 0    24907
 1     3801
 2     3091
-1     2136
 3      487
Name: drinks_code, dtype: int64
```

```
df_class.smokes_code.value_counts()

0    27538
1     4427
2     2457
Name: smokes_code, dtype: int64
```

SLIDESMANIA.COM

4. Augment the Data

Classification Model

Unneeded columns were dropped and rows with NaN values were removed. This left a dataframe with 34,422 rows and 11 columns.

```
df_class.sample(5, random_state=40)
```

	pets	body_type	height	drinks	smokes	sex	sex_code	pets_code	body_type_code	drinks_code	smokes_code
38235	likes dogs and likes cats	curvy	60.0	socially	no	f	0	1	2	0	0
56177	likes dogs	thin	62.0	socially	no	f	0	0	-1	0	0
55224	likes dogs	fit	68.0	socially	no	m	1	0	1	0	0
27481	likes dogs and likes cats	skinny	70.0	often	when drinking	m	1	1	-1	2	1
59485	has cats	average	77.0	often	yes	f	0	2	0	2	2

SLIDESMANIA.COM

4. Augment the Data (Regression Model)

New column created: diet_code

Mapped user diet preference, ie “vegetarian”, “vegan”, etc. as numerical values.

New column created: drugs_code

Mapped user responses for drug use as numerical values.

New column created: sex_code

Mapped male users as 1 and female users as 0.

```
df_reg.diet_code.value_counts()

1    20617
2     3988
0     1313
3      149
Name: diet_code, dtype: int64
```

```
df_reg.drugs_code.value_counts()

0    21041
1     4758
2      268
Name: drugs_code, dtype: int64
```

```
df_reg.sex_code.value_counts()

1    15512
0    10555
Name: sex_code, dtype: int64
```

SLIDESMANIA.COM

4. Augment the Data

Regression Model

Unneeded columns were dropped and rows with NaN values were removed. This left a dataframe with 26,067 rows and 11 columns.

```
df_reg.sample(5, random_state=4)
```

	diet	drinks	drugs	smokes	sex	height	diet_code	drugs_code	sex_code	drinks_code	smokes_code
28848	mostly anything	socially	never	no	m	72.0	1	0	1	1	0
45549	anything	socially	never	no	m	75.0	1	0	1	1	0
53378	kosher	socially	never	no	f	65.0	3	0	0	1	0
31109	anything	socially	never	no	m	68.0	1	0	1	1	0
53849	mostly vegetarian	socially	never	no	f	67.0	2	0	0	1	0

SLIDESMANIA.COM

5. Compare Classification Models

SLIDESMANIA.COM

5. Compare Classification Models

I used **Logistic Regression** and **K Nearest Neighbors (K=5)** for the classification models.

Both returned a scoring accuracy of approximately 83% on the training set .

Logistic Regression runs notably faster: 62 ms versus 198 ms for KNN (K=5).

SLIDESMANIA.COM

6. Compare Regression Models

SLIDESMANIA.COM

6. Compare Regression Models

I used **Multiple Linear Regression** and **K Neighbors Regressor (K=5)** for the regression models.

MLR returned a coefficient of determination (R^2) of 44% on the training set. KNR (K=5) scored 40%

MLR runs notably faster: 11 ms versus 253 ms for KNR (K=5).

SLIDESMANIA.COM

7. Analyze Model Metrics

SLIDESMANIA.COM

7. Analyze Model Metrics: Classification Models

For model scoring I used scikit learn's Train Test Split utility to create a testing set consisting of 25% of the available data.

K Nearest Neighbors K=5

Accuracy: 0.818
Precision: 0.817
Recall: 0.818

Logistic Regression

Accuracy: 0.827
Precision: 0.826
Recall: 0.827

The **null accuracy** [score from choosing the most common class] is 58% so the models each did pretty well, comparatively.

Null Accuracy

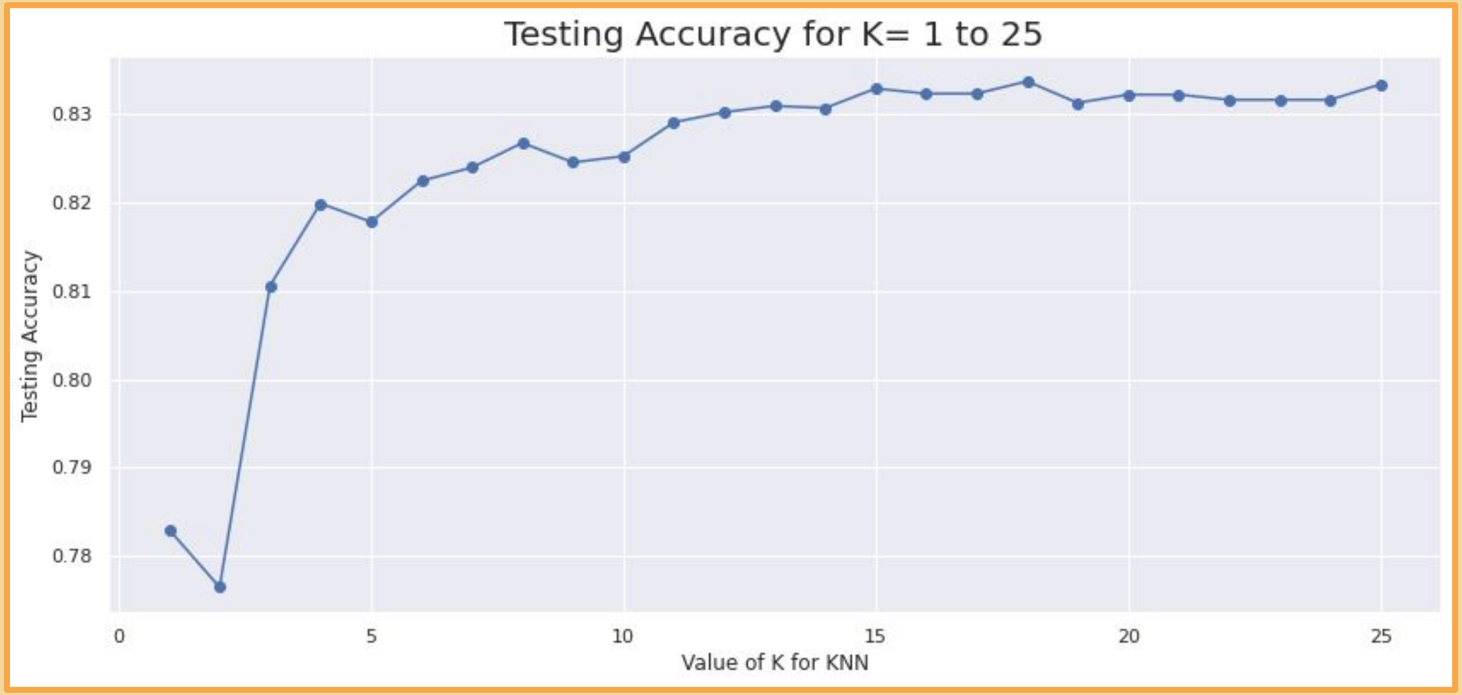
```
# Null Accuracy
y.value_counts(normalize=True)
```

```
1    0.578119
0    0.421881
Name: sex_code, dtype: float64
```

SLIDESMANIA.COM

7. Analyze Model Metrics: Classification Models

I checked to see if I could find a better value for K in the KNN model.



K=18 obtained a slightly higher model accuracy score of 0.834 around 1.6 percentage points higher than K=5.

SLIDESMANIA.COM

7. Analyze Model Metrics: Regression Models

For model scoring I used scikit learn's Train Test Split utility to create a testing set consisting of 25% of the available data.

Multiple Linear Regression

Training set R^2: 0.440
Testing set R^2: 0.449
RMSE: 2.916

K Neighbors Regressor

Training set R^2: 0.404
Testing set R^2: 0.413
RMSE: 2.929

The coefficient of determination [R^2] for both models show some correlation between the feature class and the response class, height.

RMSE of approx. **2.9** indicates error is within three units of the y axis, or in this case, inches.

Note:

The project requirements called for accuracy and precision scores for regression models, however:

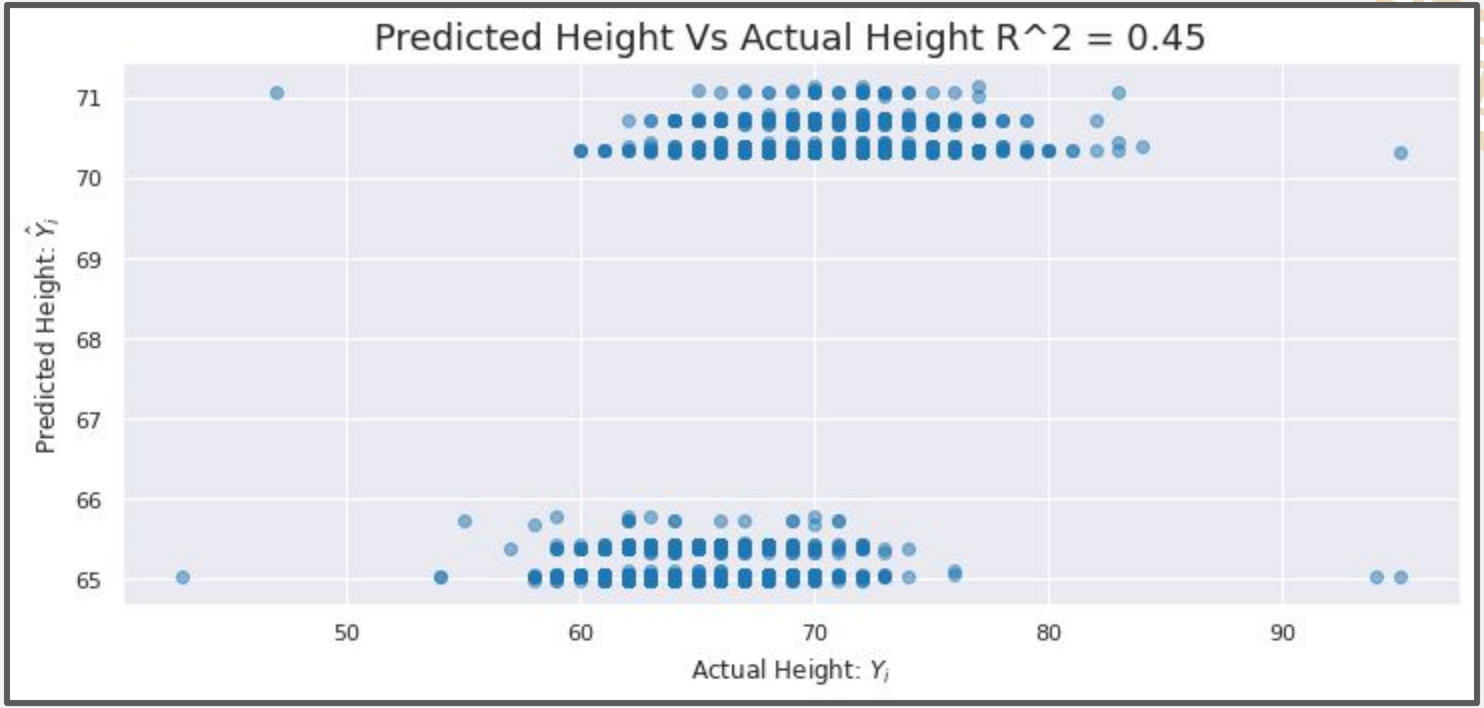
"Evaluation metrics for classification problems, such as **accuracy**, are not useful for regression problems. Instead, we need evaluation metrics designed for comparing continuous values, like the Root Mean Squared Error, or **RMSE**."

Source: *DataSchool*

SLIDESMANIA.COM

7. Analyze Model Metrics: Regression Models

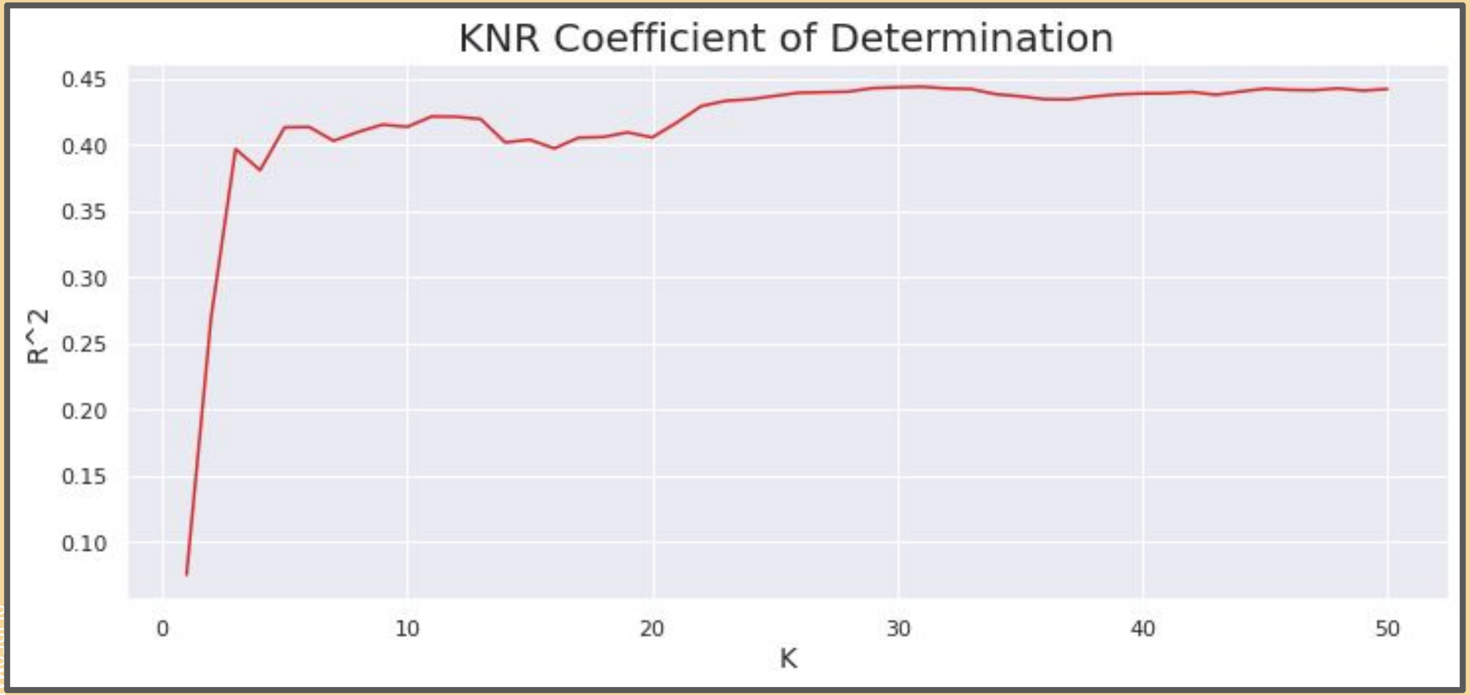
Multiple Linear Regression



SLIDESMANIA.COM

7. Analyze Model Metrics: Regression Models

I checked to see if I could find a better value for K in the KNR model.



K=31 obtained a slightly higher model R^2 score of 0.444 only 3.1 percentage points higher than K=5.

SLIDESMANIA.COM

8. Conclusion

SLIDESMANIA.COM

8. Conclusion

Bottom line is that I was able to construct two classification models that were decently predictive and two regression models that showed a faint linear relationship between the feature and response classes.

The provided dataset was honestly terrible, in my opinion. This may have been intentional from a teaching standpoint, but it wasn't very conducive for machine learning.

As it was data from ≅60k users on a dating site, it's possible none of the data was truthful. This was apparent looking at the income values (surprising number of millionaires!) and reported heights (amazing number of people under 4-feet and over 7-feet tall). Shocking it included responses from two users over the age of 108! And finally, the **10** essay columns were utterly useless and made up ≅30% of the dataset.

In the end I did learn a lot about cleaning and mapping data for machine learning. I just wish the data provided would have had greater validity and been more interesting to construct machine learning models with.

SLIDESMANIA.COM