```
import warnings

warnings.filterwarnings('ignore')

import statsmodels.api as sm
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score, accuracy_score

from sklearn.datasets import load_iris
from sklearn import tree

from sklearn.tree import DecisionTreeClassifier # Decision Tree

%matplotlib inline
from plotnine import *
from sklearn.decomposition import PCA
import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
import numpy as nps
from scipy import stats
from sklearn.cluster import KMeans

from sklearn.model_selection import KFold

from matplotlib import pyplot as plt
from sklearn.metrics import silhouette_score
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import plot_confusion_matrix

import numpy as np


tuition_cost = pd.read_csv('https://raw.githubusercontent.com/rfordatascience/tidytues
student_diversity = pd.read_csv('https://raw.githubusercontent.com/rfordatascience/tid
salary_potential = pd.read_csv('https://raw.githubusercontent.com/rfordatascience/tidy


#Renamed INSTITUTION in student diversity table to name
student_diversity.rename(columns = {"INSTITUTION": "name"},
          inplace = True)


#------DATA CLEANING ------------------
colleges = salary_potential['name']
for college in colleges:
  #If there are parentheses
  if '-' in college:
```

```
      college = college.split('-',1)[0]


  #Merge the tuition cost and salary potential dataframes
  tuition_salary = pd.merge(tuition_cost, salary_potential, on='name')

  #Drops the null values that did not match from the datasets
  tuition_salary.isnull().sum()
  tuition_salary = tuition_salary.dropna()

  #----- DATA CLEANING -----------

  #Need to relace U. with University and then remove whatever
  #Need to either, remove the substring after "College"
  #Somehow get the substring of the state and then remove the state form that.


  #Create a list of the college names. Then iterate through and clean
  #Then add it back to the dataframe
  colleges = student_diversity['name']
  colleges_clean = []

  for college in colleges:

    #If the college starts with U.
    if college[0] == "U" and college[1] == '.':
      college = college.replace('U.', 'University')

    #If the college ends with U., we remove what comes after
    if 'U.' in college:
      college = college.split('U.',1)[0] + 'University'

    #If the college ends with College we remove what comes after
    if 'College' in college:
      college = college.split('College',1)[0] + "College"

    #If there are parentheses
    if '(' in college:
      college = college.split('(',1)[0]

    #Itetate through string, current letter is upopercasem the prev leetter was lowercas
    #Then remove everyting uppercase and forward
    for i in range(1,len(college)):
      if college[i-1].islower() and college[i].isupper():
        college = college[:i]
        break

    #Append to new list
    colleges_clean.append(college)
    # print(college)
```

```
#-----------------------


#Replace the name of college with the cleaned names
student_diversity['name'] = colleges_clean
# print(colleges_clean)


#Final merge
df = tuition_salary.merge(student_diversity, on='name')


#Cleaning Variable Names
df.columns= df.columns.str.strip().str.lower()
df.columns= df.columns.str.replace(" ", "_")
df.columns= df.columns.str.replace("/", "")

df["enrollment"] = df["enrollment"].str.replace(",","")
df["total_minority"] = df["total_minority"].str.replace(",","")

df["enrollment"] = pd.to_numeric(df["enrollment"])
df["total_minority"] = pd.to_numeric(df["total_minority"])

df["minority_percentage"] = df["total_minority"] / df["enrollment"]
df["avg_tuition"] = (df["in_state_total"] + df["out_of_state_total"] ) / 2


#Adding a regions column

#ADDING REGIONS
west = ["CA","ID","MT","WY","NV","UT","CO","AZ", "NM", "AK", "WA", "OR"]
northeast = ["ME","NH","VT","MA","RI","CT","NY","PA","NJ"]
south = ["DE","MD","VA","WV","NC","SC","GA","FL","KY","TN","MS", "AL", "OK","TX","AR",
midwest = ["WI","MI","IL","IN","OH","ND","SD","NE","KY","MN","IA","MO"]

df['state_code']= df['state_code'].astype(str)

df.dtypes

condition = [(df['state_code'].isin(west)),
             (df['state_code'].isin(northeast)),
             (df['state_code'].isin(south)),
             (df['state_code'].isin(midwest))]

values = ["West", "Northeast", "South", "Midwest"]

df['region'] = np.select(condition, values)



df.head()
```

```
# df.head()

#Map of all the states
#Check from the back and then remove if its exist
```

	name	state	state_code	type	degree_length	room_and_board	in_
0	Adams State University	Colorado	CO	Public	4 Year	8782.0	
1	Agnes Scott College	Georgia	GA	Private	4 Year	12330.0	
2	Alabama State University	Alabama	AL	Public	4 Year	5422.0	
3	Alaska Pacific University	Alaska	AK	Private	4 Year	7300.0	
4	Albertus Magnus College	Connecticut	CT	Private	4 Year	13200.0	

```
# df['asian']=df['asian'].astype(str).str.replace(',', '').astype(int)
# df['white']=df['white'].astype(str).str.replace(',', '').astype(int)
# df['black']=df['black'].astype(str).str.replace(',', '').astype(int)

# df3 = df.groupby(['type'], as_index=False).sum()
# df3.head()

# race = df3[['type','asian','black','white']]
# race.head()
# # race.head()

# # Very simple one-liner using our agg_tips DataFrame.
# race.plot(kind='bar', stacked=True)
```

```
# # Just add a title and rotate the x-axis labels to be horizontal.
# plt.title('Tips by Day and Gender')
# plt.xticks(rotation=0, ha='center')
```

Kai Itokazu

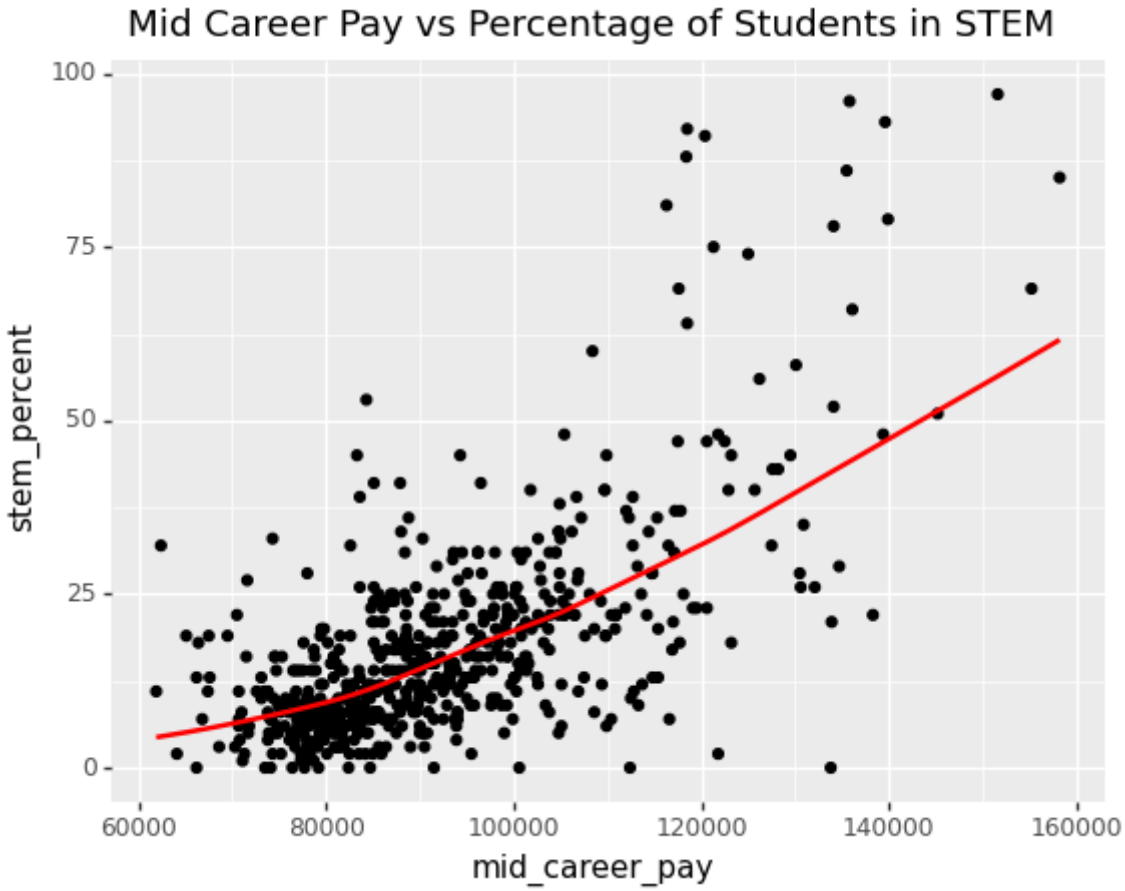
Question 1: Correlation between the average percentage of students in STEM, and early career pay or mid career pay? (Supervised)

- A. We are going to use model validation and linear regression models to see the significance being in STEM has for early career pay and mid career pay. We can also perform a hypothesis test and see if the independent variables are significant enough to reject the null hypothesis, that STEM has no effect on career pay. **We can also create a variation of the question and use percentage of students in STEM, early career pay, location, and make the world better percent as predictors to predict the mid career pay. We could use cross validation for this.**
- B. We believe that estimating salary is a good dependent variable. There are also good predictor variables such as students in stem that we can leverage to create predictions. We can make use of cross validation to assess the accuracy of the models.
- C. To create a better visual representation, we can use simple bar graphs to plot the mean early/mid career pay for the majors in STEM and the majors that are not in STEM. If we are going with the direction of the model training, we can plot a confusion matrix.

- a) use an 80/20 train test split for model validation and make sure you z score your continuous variables
- b) check the linearity assumption for your continuous variables using ggplot. Discuss in detail what you are checking for and specifically what you see for this model.
- c) check heteroskedasticity by plotting predicted reaction times/residuals using ggplot. Discuss in detail what you are checking for and what you see for this model.
- d) plot the actual vs. predicted reaction times, as well as print out the mean absolute error and R^2 for your model for both train and test. How well did your model do based on these metrics, and how can you tell?
- e) is your model overfit? How can you tell?
- f) make a bar chart showing the coefficient values (x should be each coef name, the height of each bar should be the value of the coefficient).

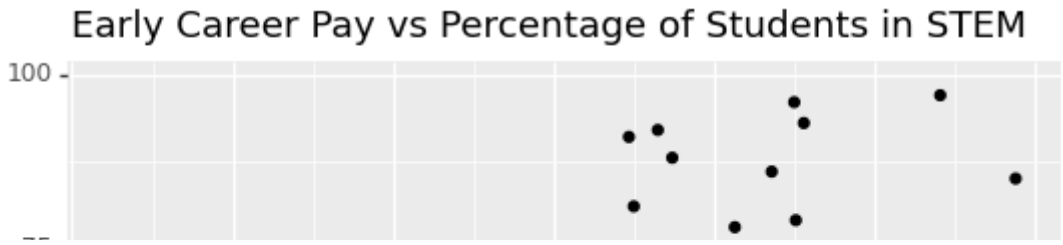
Here the scatterplot shows a correlation between mid career pay and the percetnage or students

```
ggplot(df) + aes(x='mid_career_pay',y='stem_percent') + geom_point() + geom_smooth(col
```




```
<ggplot: (8771369039085)>
```

```
ggplot(df) + aes(x='early_career_pay',y='stem_percent') + geom_point() + geom_smooth(c
```



Here we plotted the mid career pay with the percentage of students in stem. As you can see there is definitely a correlation between the two variables. We also plotted the ealry career pay with the percentage of students in stem. We can see that there is a positive correlation for both of these variables. I wanted to plot whether these were correlated before running the linear regression model. There are studies that show that for entry level positions, stem careers tend to have a higher salary than other careers. Obviously salary is not everything, and it is highyl affected by the location. However, from the data given to us, we see that there is a noticeable correlation between the two variables

```
# Split Data
# Put more variables that may affect career pay into the TTS
genderDUM = pd.get_dummies(df["region"])

df = pd.concat([df, genderDUM], axis = 1)

df["avg_career_pay"] = (df['early_career_pay'] + df["mid_career_pay"]) / 2

df.head()

predictors = ["stem_percent", "make_world_better_percent", "West", "Northeast", "South

X_train, X_test, y_train, y_test = train_test_split(df[predictors], df["avg_career_pay
```

We are going to run a linear regression model and use percentage of students enrolled in STEM, make world better percent, and region as predictors to predict the average career pay. We created a new column called avg career pay which averages out the early career pay and the mid career pay. We did this because we wanted to predict the overall relationship between someone's salary with out predictors; and we thought the best way to do this with the data we have is to average out the early career salary and the mid career salary. Because the predictors are already in terms of percentage, we do not have to Zscore the variables.

```
model = LinearRegression()
model.fit(X_train, y_train)
```

```
LinearRegression()
```

```
# predictions
y_pred = model.predict(X_test)
y_pred[1:10]
```

```
array([63445.24913485, 66480.83257042, 80504.83225597, 84697.30501094,
       64684.56224339, 71136.04444616, 90900.79297028, 63289.03704316,
       62667.64988473])
```

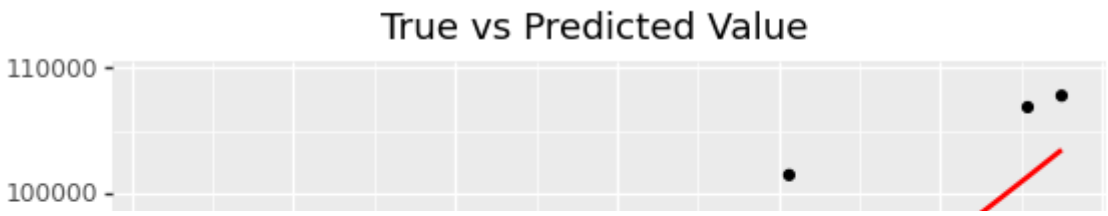
```
testr2 = model.score(X_test, y_test) #testing R2
```

```
trainr2 = model.score(X_train, y_train) #training R2
```

```
true_vs_pred = pd.DataFrame({"predict": y_pred,"trueV": y_test})
true_vs_pred.head()
```

	predict	trueV
585	62976.612860	61350.0
448	63445.249135	70450.0
107	66480.832570	58750.0
456	80504.832256	85500.0
59	84697.305011	84350.0

```
(ggplot(true_vs_pred, aes(x = "trueV", y = "predict")) + geom_point() + geom_smooth(co
```




```
print("Training R2: ", testr2)
print("Testing R2: ", trainr2)
```

```
Training R2:  0.626194143373814
Testing R2:  0.6182493775612414
```

After using a train test split on our linear regression model, we got a training r2 of 0.63 and a testing r2 of 0.62. For traning, we can say that 0.62 indicates that 62% of the variability in the outcome data can be explained by the model and for the testing we can say that 63% of the variability in the outcome data can be explained by the model. I would say this shows a mid-high level correlation between the variables. The predictors I chose also correlate to the salary pay

K FOLD CROSS VALIDATION (Using 10 Splits)

```
kf = KFold(n_splits = 10)

X = df[predictors]
y = df["mid_career_pay"]

lr = LinearRegression()
accuracy = []
mse_test = []
mse_train = []
mse = []
r2 = []


for train,test in kf.split(X):
    X_train = X.iloc[train]
    X_test  = X.iloc[test]
    y_train = y[train]
    y_test  = y[test]

    # model
    model = lr.fit(X_train, y_train)
    # record accuracy
    mse.append(mean_squared_error(y_test, model.predict(X_test)))
    r2.append(r2_score(y_test,model.predict(X_test)))

mse_test.append(mean_squared_error(y_test,model.predict(X_test)))
```

```
mse_train.append(mean_squared_error(y_train,model.predict(X_train)))
r2.append(r2_score(y_test,model.predict(X_test)))

print("Mse test: ", mse_test[0].round(3))
print("Mse train: ", mse_train[0].round(3))
print("r2: ", np.mean(r2).round(3))
```

```
Mse test:  123435111.519
Mse train:  89761914.406
r2:  0.582
```

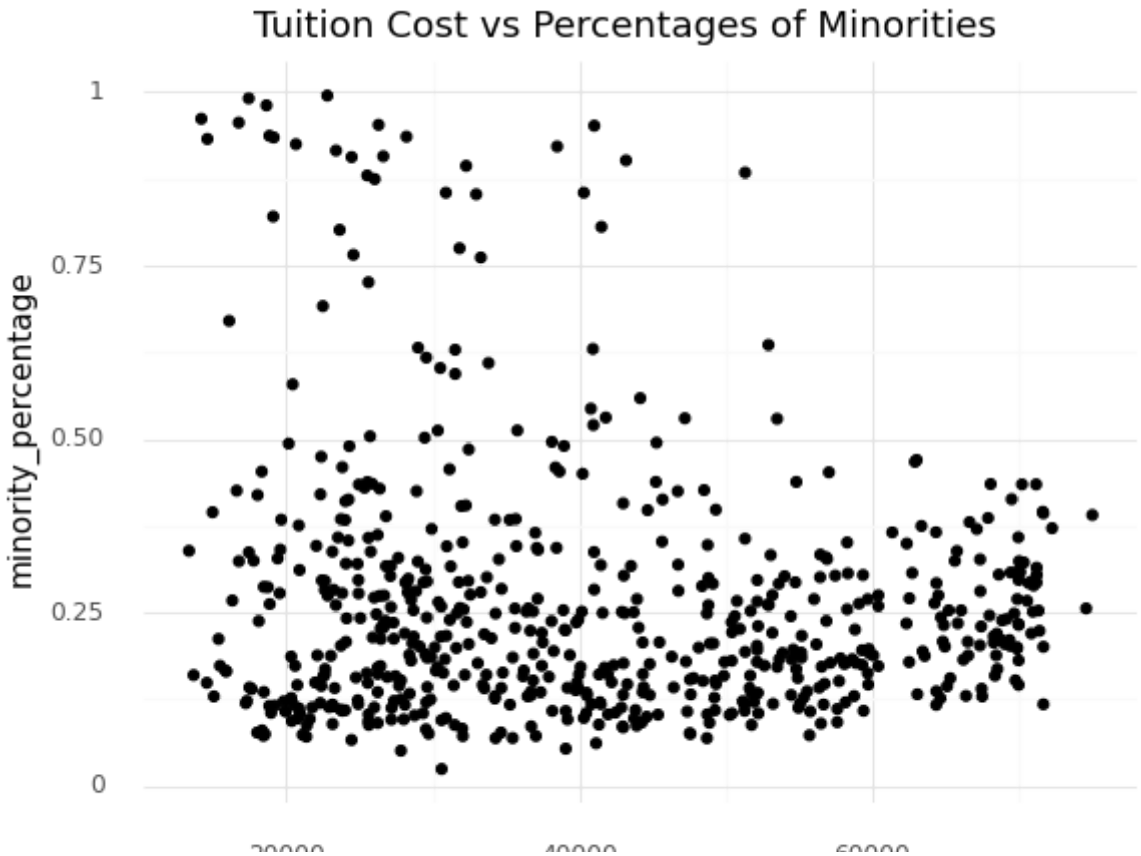
Here we have the MSE train and testing shown above. Obviously it is a little hard to understand the MSE without understanding the context of the problem. However, we can see that the testing MSE is greater than the training MSE that indicates that are model is overfit. This means that are model cannot perform as accurately against unseen data.

We see that from the K fold split, we got a R2 of 0.582. This shows that 58% of the variability can be explained from our model. This does not indicate that the variables are highly correlated. But there is a correlation that is present.

Question 2: Does the number of students enrolled, race, and overall tuition vary depending on the type of school they attend (public, private). (find continuous variable and cluster and describe them)

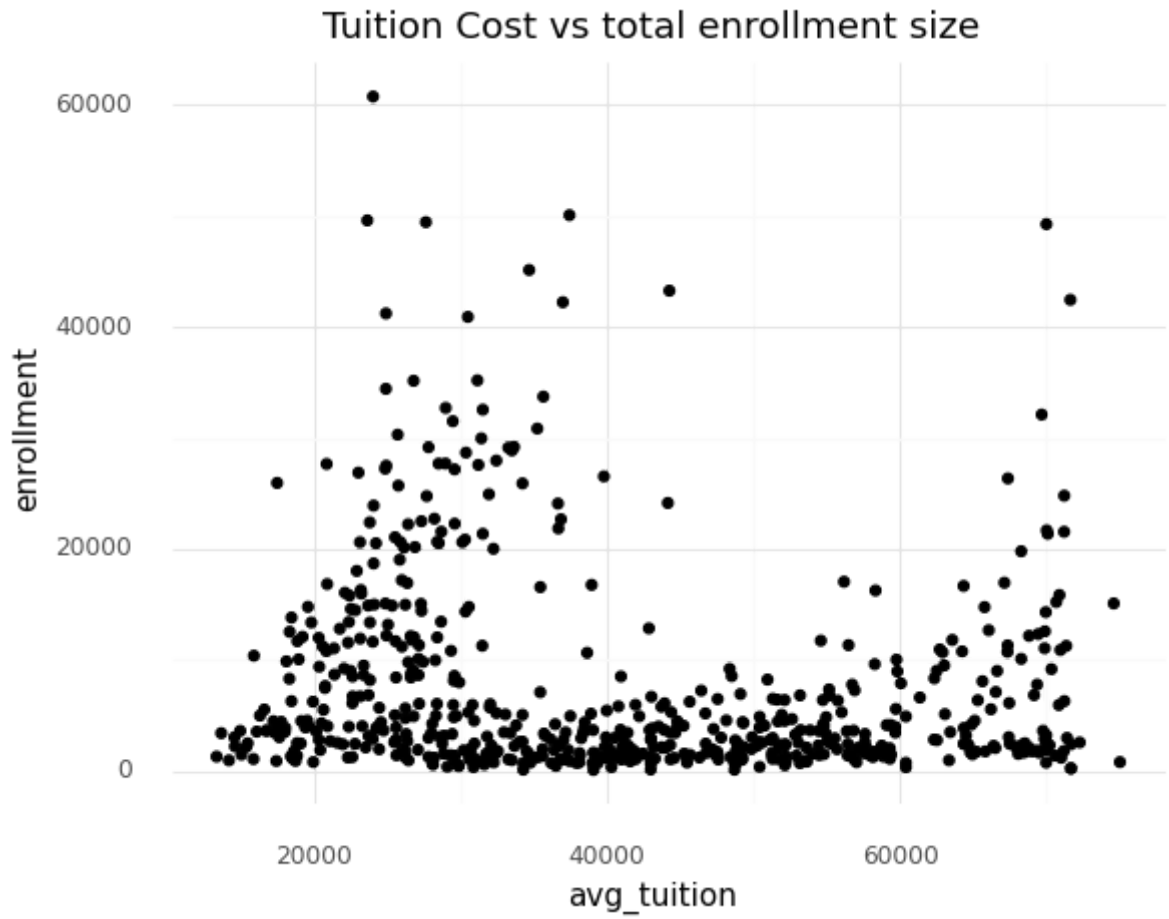
- A. We will use K-means and test model validation to cluster the variables by schools they attend (public, prive, or for profit). We will choose K random points to be clustering centers. For each K point, points will be clustered to the K point that it is closest to. Then you would recalculate the center until it is stable or converges.
- B. In K-means, we are trying to find different groups that have similar traits and assign them to clusters. In this model, we are trying to use the variables and cluster them into the different types of schools to see what is the likelihood of these variables playing a significant role in school type.
- C. We can use geom_point() to create a scatter plot based on the variables that we are using as predictors for the clustering. We can also create a confusion matrix to check the performance of our classification model.

```
(ggplot(df, aes(x = "avg_tuition", y = "minority_percentage")) + geom_point() + theme_
+ ggtitle("Tuition Cost vs Percentages of Minorities"))
```




```
(ggplot(df, aes(x = "enrollment", y = "minority_percentage")) + geom_point() + theme_r
+ggtitle("Enrollment vs Percentages of Minorities"))
```

```
(ggplot(df, aes(x = "avg_tuition", y = "enrollment")) + geom_point() + theme_minimal()
+ggtitle("Tuition Cost vs total enrollment size"))
```

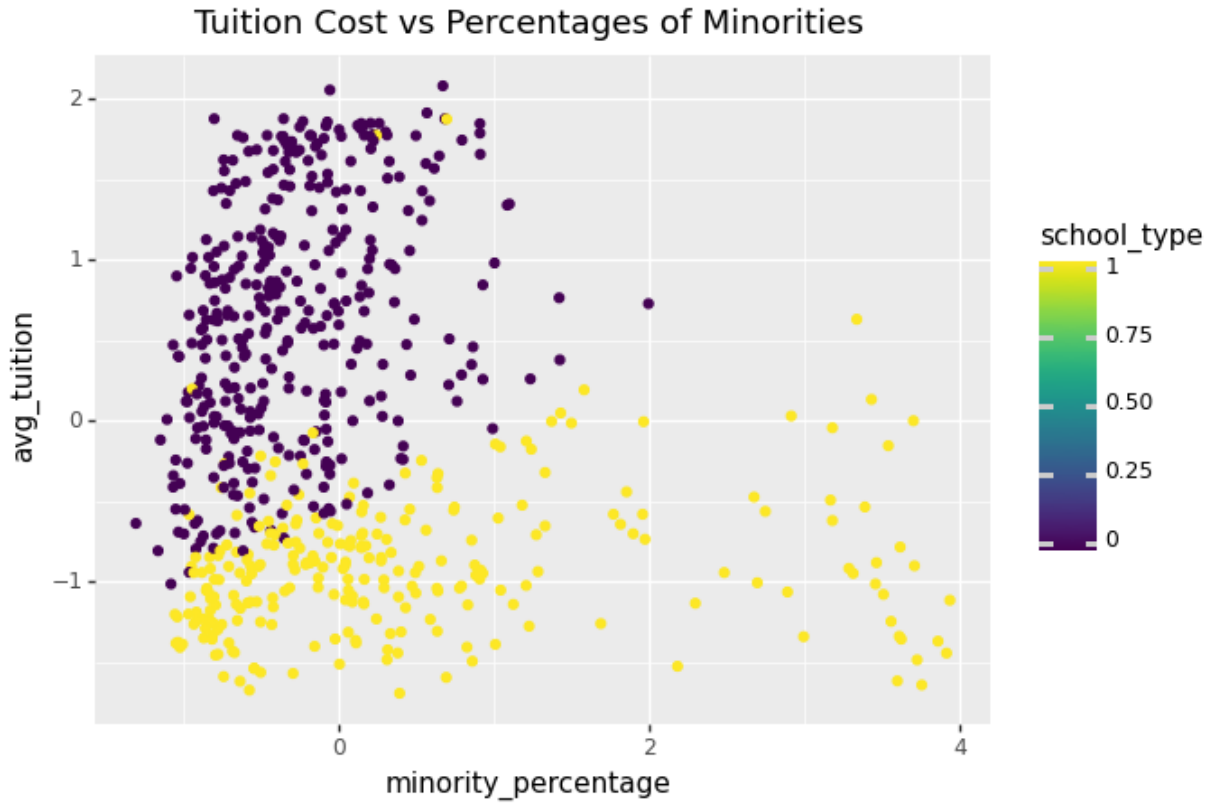



```
<ggplot: (8771365097085)>
```

We creating a clutstering alogrithm, assuming 2 clusters. One representing private schools and other representing public schools. Looking at the graphs, there seems to be 2 distinct clusters. We want to consider factors such as shape, density, separation, cohesion and noise.

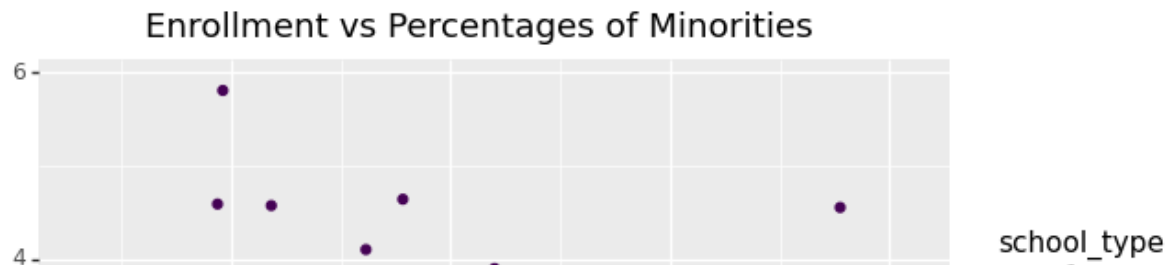
```
features = ["minority_percentage", "avg_tuition","enrollment"]
X = df[features]

z = StandardScaler()
X[["minority_percentage", "avg_tuition","enrollment"]] = z.fit_transform(X)

km = KMeans(n_clusters = 2)
km.fit(X)

school_type = km.predict(X)

X["cluster"] = school_type
```

```
(ggplot(X, aes("minority_percentage", "avg_tuition", color = "school_type")) + geom_pc
```

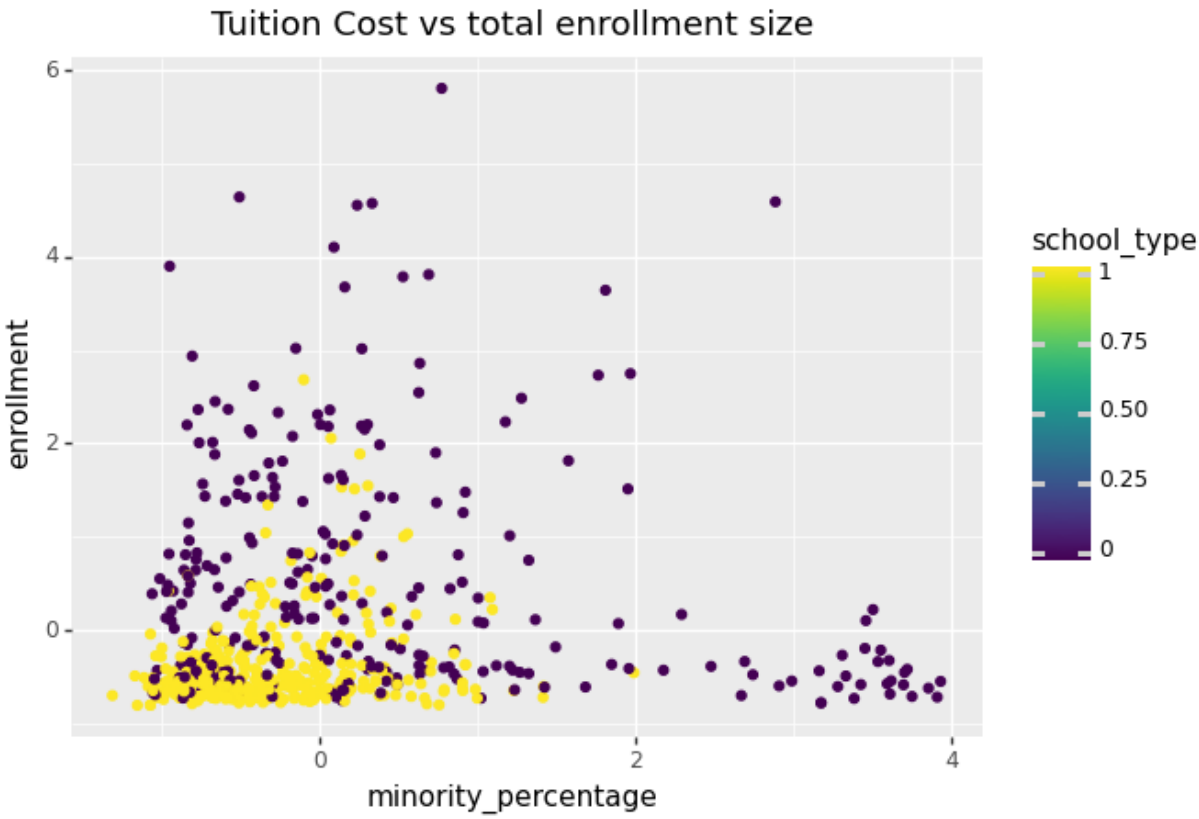



```
<ggplot: (8771360200853)>
```

```
(ggplot(X, aes("avg_tuition", "enrollment", color = "school_type")) + geom_point() + g
```




```
(ggplot(X, aes("minority_percentage", "enrollment", color = "school_type")) + geom_poi
```




```
<ggplot: (8771367116077)>
```

```
silhouette_score(X[features], school_type)
```

```
0.341034024620713
```

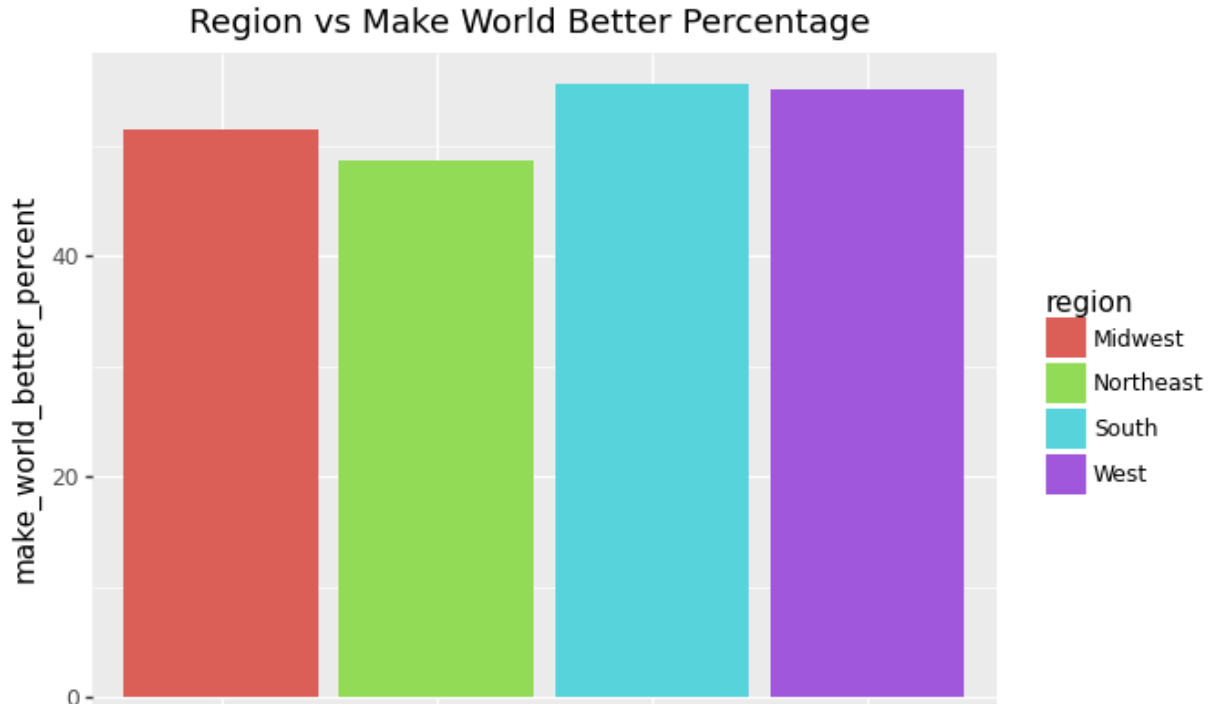
K means makes assumptions about the shape of spherical clusters. We also standardized the variables to make sure that all of the variables are on the same scale. This is important with K means. Also another assumption where there is roughly the same number of points in each cluster. Looking at the first plot, we can see that there are 2 distinct clusters in the tuition cost and percentage of minorities. We know for a fact that private schools have a higher average tuition cost. The purple cluster appears to be representative of the private schools while the yellow clusters appear to be representative of public schools. It is also very interesting to see how the clusters are

grouped in terms of minority percentage. While the yellow clusters have a range of various minority percentages, the purple clusters have a significantly lower percentage, being around zscore of 0 to -1. This shows that public schools tend to have a higher percentage of minorities which make sense. There can be an ethical statement about the racial diversity in private institutions. While most private schools offer scholarship money, education experts say the cost of private schools is often the largest barrier to creating more racial and ethnic diversity. With enrollment vs percentage of miorities, there are clusters that are divided by the average tuition. These clusters are pretty dense. Again, this makes sense when we are considering the cost of private vs public schools. And then we can also in the third plot we see that the clusters are not very separated. However, we know from before that it makes sense when we see the yellow cluster as private schools and purple cluster as public schools. We see that the yellow clusters have high density. We however, cannot conclude that just because private schools have a lower minority percentage, that all public schools hae a high minority percentage. There are a lot of other factors to consider such as location that play a high factor in terms of the diversity of an institution.

Question 3: Is there a relationship between the state of school and the number of alumni who think they are making the world a “better place”? (just use plots)

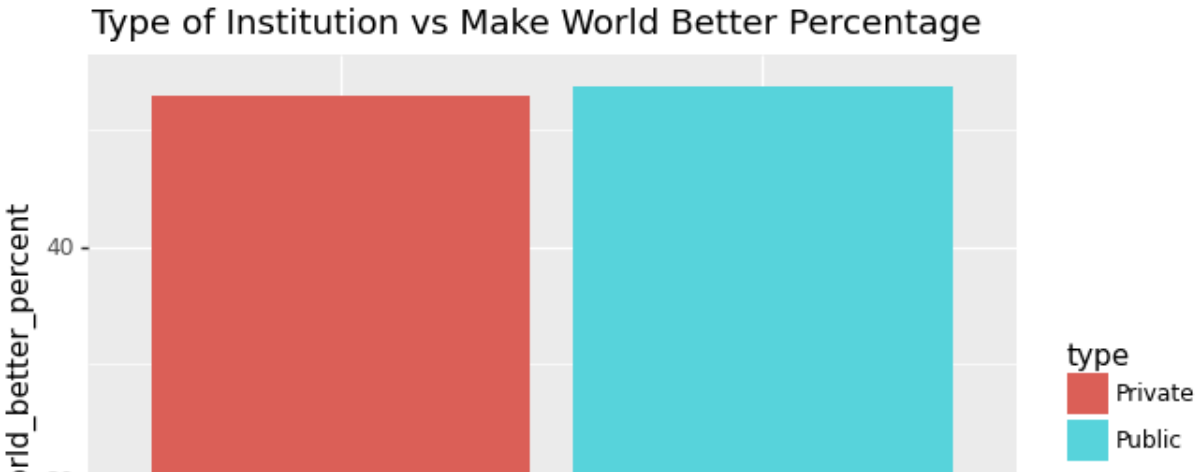
- A. We can modify this question slightly by creating a model to see whether more than half of the alumni consider their school/education to be making the world a better place based on specific predictors. We can utilize continuous factors such as percentage of minorities and tuition.
- B. We can use a KNN model to see whether alumni consider the school to be ranked as “making the world a better place”. We can use a grid search to select our hyper parameter to select which K to use.
- C. We can use plotKNN to plot the KNN model as well as a confusion matrix to plot the testing models. (one more plot)

```
df3 = df.groupby(['region'], as_index=False).mean()

df3 = df3.drop(0)
df3.head()

ggplot(df3) + aes(x='region',y='make_world_better_percent', fill="region") +  geom_col
```



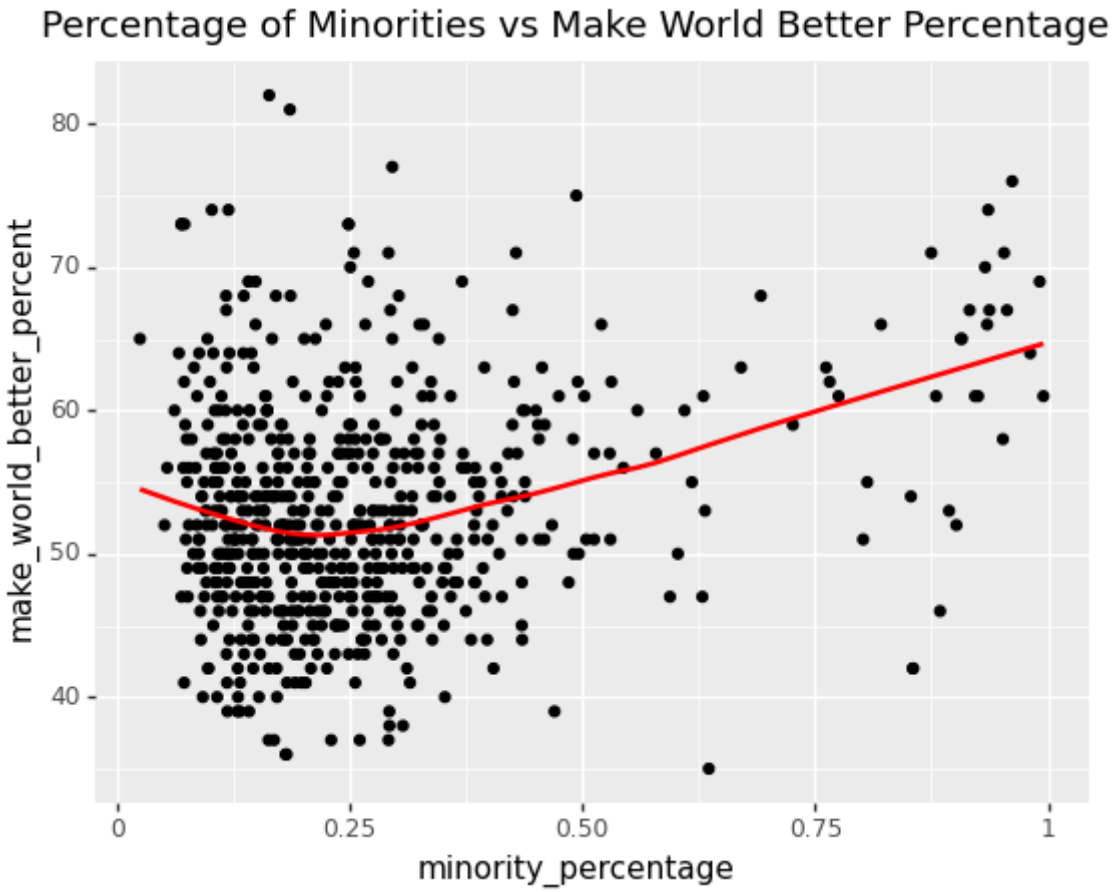
Here we have a box plot that shows the schools from different regions against the average make a world better percentage. Although the results are fairly similar though each region, we see that the schools in the South on average, have the most people reporting back that they feel they are making the world a better place. The next is the schools in the west, then midwest, and then last the northeast. This is hard to conclude whether these regions and schools make a difference since there are a lot of different factors for each school, and the variable is very subjective. This is what the alumni think whether they are not they are contributing to the world's goodness.

```
df5 = df.groupby(['type'], as_index=False).mean()

df5 = df5.drop(0)
df5.head()

ggplot(df5) + aes(x='type',y='make_world_better_percent', fill="type") +  geom_col() +
```



Here we see the average make a world better percent with the type of school, public or private. Although it is higher for public schools, they are very close

```
ggplot(df) + aes(x='minority_percentage',y='make_world_better_percent') +  geom_point(
```

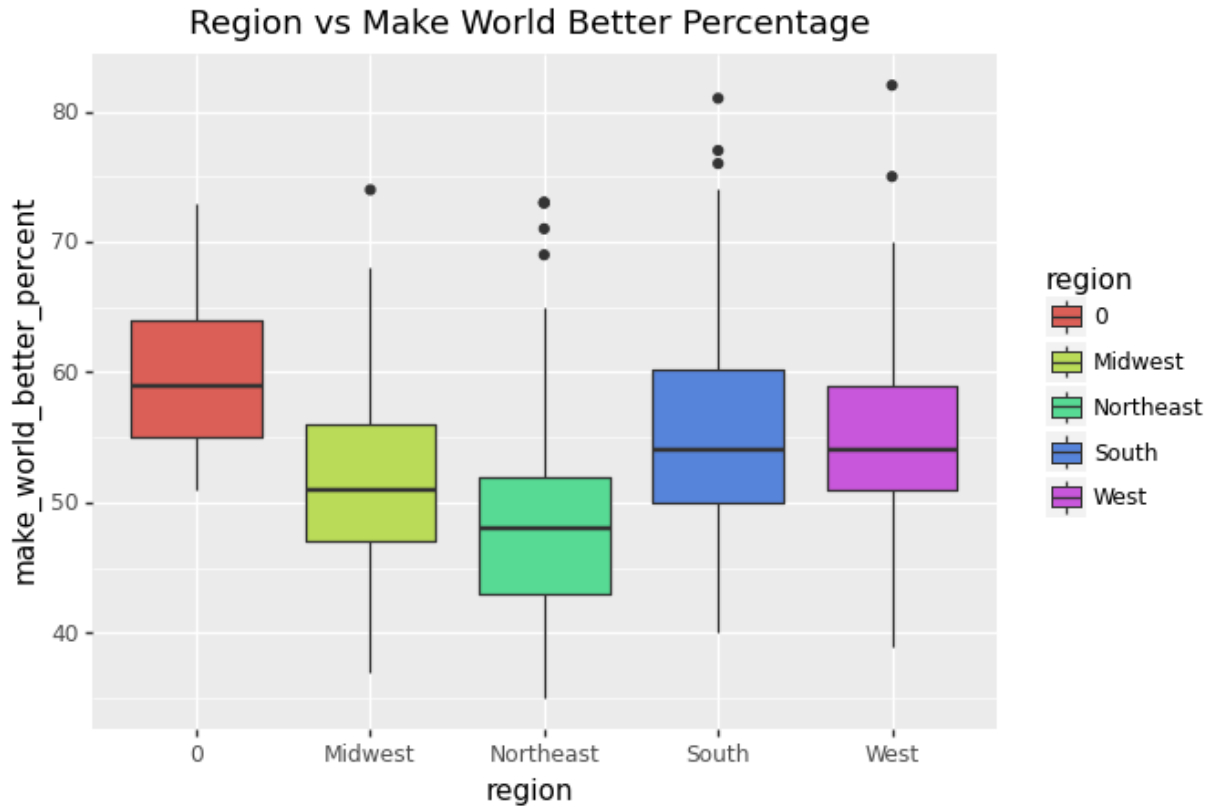



```
<ggplot: (8771365745289)>
```

The scatterplot above suggests that there may be a linear correlation between the percentage of minroties at the school veresus the amount of students that feel they are making the world a better

place. We cannot conclude anything from the scatters due to the lack of data points but it is

```
ggplot(df) + aes(x='region',y='make_world_better_percent', fill="region") +  geom_boxp
```




```
<ggplot: (8771366789585)>
```

(Ignore the 0 box plot) Although we saw from the bar charts that on average, the MWBP(make world better percentage) is fairly similar through each region, the boxplot displays the differences a little better. The IQR for the South adn West are much higher than Midwest and Northeast. Furthermore, there are outliers that are very high for the South and West.

```
df4 = df[['minority_percentage','make_world_better_percent','avg_tuition','enrollment'

sns.heatmap(df4.corr(),cmap="coolwarm",annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa3e2df1690>
```

The heatmap above shows the correlation between various variables. The one I used were minority percentage, make world better percentage, average tuition, enrollment, and mid career pay. The things that stand is the orrelation between average tuition and mic_career_pay. It seems that there is a positive correlation between the two variables. There is also a slight negative correlation between make world better percent and average tuition, and make world better percent and mid career pay. This suggests that from the data,assuming that a high tuition indicates a private school, private schools might have on average higher mid career pay from the students. There are obviiously a lot of other factors to consider, but this is what we can conlcude from the heatmap.

Cristin Lue

Question 1: What is the average tuition cost for a student applying to a public school given that they are in state vs out of state.

```
public_school = df["type"] == "Public"
df.loc[public_school]

public_school_in_state = (df.loc[public_school, "in_state_tuition"].mean())
public_school_out_state = (df.loc[public_school, "out_of_state_tuition"].mean())
print("The average tuition cost for a student applying to a public school given that t
      public_school_in_state)
print("The average tuition cost for a student applying to a public school given that t
      public_school_out_state)
```

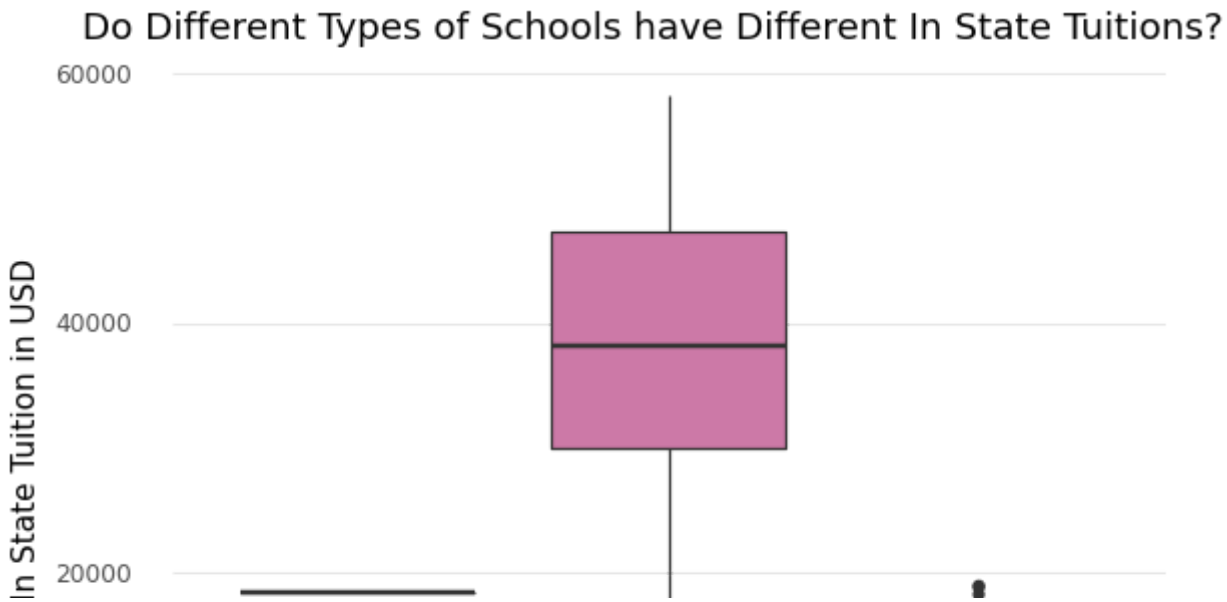
```
The average tuition cost for a student applying to a public school given that th
The average tuition cost for a student applying to a public school given that th
```

We kept the first part of calculating the mean tuition cost for students who attend public school and compare the difference given that the public school the student decides to attend is in-state or out-of-state. However, we decided to delete the idea of using the Naive Bayes algorithm. We decided we do not need a classification algorithm since we are simply just calculating the average tuition cost for public schools given that they are in-state vs out-of-state.

We first located only public schools in the data frame. Then, from there we calculated the in-state tuition and out-of-state tuition means. This answer tells us that people who attend public schools out-of-state usually pay more for their tuition (9,697.32 USD) than people who attend public schools in-state (21,607.81 USD). But from the graphs below we can see that on average, students who attend private schools will always pay more than than attending public schools, no matter if they pay in-state or out-of-state tuition. This makes sense because people who usually pay more to attend schools that are out-of-state than in-state are non-resident students' who come from families who don't pay taxes to the school's state so it's accounted for in their tuition.

Instead of creating a bar graph and a confusion matrix, we decided to create two different box plots. The first graph compares the in-state tuition given the different type of schools and the second graph compares the out-of-state tuition given the different type of schools. We changed from a bar graph to a box plot because with a box plot we can easily visualize the difference in the median tuition between the different types of school and the distribution of our data (whether the data set is spread out or dense). We deleted the bar graph because we realized that bar graphs are not a good representation of our question. We also cannot accurately represent average tuition cost using a bar graph. Additionally, we deleted the confusion matrix because we did not use a model to predict whether a student is in-state vs out-of-state given that they go to a public school

```
# box plot
    # comparing type of school VS in-state tuition
(ggplot(df, aes(x = "type", y = "in_state_tuition", fill = "type")) +
 geom_boxplot() +
 theme_minimal() +
 ggtitle("Do Different Types of Schools have Different In State Tuitions?") +
 labs(x = "", y = "In State Tuition in USD") +
 scale_fill_manual(["#d55e00", "#cc79a7", "#0072b2","#f0e442","#009e73"]) +
 theme(panel_grid_major_x = element_blank(),
       panel_grid_minor_y = element_blank(),
       legend_position = "none"))
```



For our first plot we created a box plot comparing the different types of schools to see if there is a difference for in-state tuition. Here we can see that although the tuition cost varies for in-state tuition across all types of schools, usually private schools pay the most on average. We cannot really compare the for profit school since there is only one for profit school in the data set. Thus, if you attend the for profit school you will most likely pay more for tuition than a public school given that you are paying for in-state tuition. Here we can see that the distribution for public schools is very compact and dense with a few outliers. For public schools, we can also see that most of the data lies below the lower quartile of private schools. For private schools the distribution is very spread out and has a wide range.

```
# box plot
    # comparing type of school VS out-of-state tuition
(ggplot(df, aes(x = "type", y = "out_of_state_tuition", fill = "type")) +
 geom_boxplot() +
 theme_minimal() +
 ggtitle("Do Different Types of Schools have Different Out of State Tuitions?") +
 labs(x = "", y = "Out of State Tuition in USD") +
 scale_fill_manual(["#d55e00", "#cc79a7", "#0072b2","#f0e442","#009e73"]) +
 theme(panel_grid_major_x = element_blank(),
       panel_grid_minor_y = element_blank(),
       legend_position = "none"))
```

For our second box plot we looked at how much the different types of schools have students pay for tuition given that they are out-of-state. On the x axis we have the type of schools and on the y axis we have the out-of-state tuition in USD. Here we can see that the distributions of the 3 box plots are a lot more spread out compared to the in-state tuition. Additionally, the median is a lot higher for all 3 types of schools than in-state tuition, which generally makes sense because on average, students

who attend school out-of-state pay more for tuition than in-state. Here we can see that the box plot for public school is a lot more spread out, which means there is a lot more variety given for the out-of-state tuition given that they go to a public school. This makes sense because some states are more populated than others and thus will make their students pay more for out-of-state tuition due to rural inflation. Once again, for public schools, there are outliers for this box plot. We can see that the outliers are a lot higher compared to the outliers for the public school boxplot given that they pay for in-state tuition.

Question 2: Perform Principal Component Analysis (PCA) on in-state tuition, in-state total, out-of-state tuition, out-of-state total, room and board, stem percent, make the world a better place percent, early and mid-career pay to figure out if there's some way to combine these numeric features so that we can only use a couple principal components rather than the entire data set while still explaining as much information as possible.

We changed our question. Before it said, "How many factors play an important role between early career pay vs mid career pay and which coast (east coast or west coast) has a higher early career pay as opposed to mid career pay." Now, we changed it to, "Perform Principal Component Analysis (PCA) on in-state tuition, in-state total, out-of-state tuition, out-of-state total, room and board, stem percent, make the world a better place percent, early and mid-career pay to figure out if there's some way to combine these numeric features so that we can only use a couple principal components rather than the entire data set while still explaining as much information as possible." We changed this question so we can perform PCA on the variables that were numeric (floats and intergers). We can't perform PCA on variables that are objects or strings because PCA is a dimensionality reduction model.

```
predictors = ["in_state_tuition", "in_state_total", "out_of_state_tuition", "out_of_st

X_train, X_test, y_train, y_test = train_test_split(df[predictors],
                                                    df["rank"],
                                                    test_size = 0.2)

# z score continuous variables
z = StandardScaler()
Xz_train = z.fit_transform(X_train[predictors])
Xz_test = z.transform(X_test[predictors])

# apply PCA on the training set
pca = PCA()
pca.fit(Xz_train)
```

```
pcaDF = pd.DataFrame({"expl_var": pca.explained_variance_ratio_,
                      "pc": range(1,10),
                      "cum_var": pca.explained_variance_ratio_.cumsum()})
pcaDF.head()
```

Despite changing our question, we were still able to perform PCA. We used a train test split for model validation in order to determine how many variables we need to keep in order to still account for the majority of variation for the rank of each school. We perform model validation to make sure our model does well on data that it's never seen before and doesn't pick up on any "patterns" from our sample data that doesn't apply to our entire data set. Additionally, we decided to z-score our predictor variables so they can all be on the same scale or baseline. Here we can see in the new data frame the number of principal components (PC), the amount of variance explained per PC, and the cumulative variance as the number of PC increases. PCA is a dimensionality reduction method that helps speed up computational time. It is a good model because we are able to keep majority of the data's variance while only keeping the most important variables.

```
# scree plot
    # the amount of variance explained per PC
(ggplot(pcaDF, aes(x = "pc", y = "expl_var")) +
 geom_line() +
 geom_point() +
 theme_minimal() +
 ggtitle("The Amount of Variance Explained by Each Principal Component") +
 labs(x = "Number of Principal Components (PC)", y = "Amount of Variance Explained") +
 ylim(0,1))
```

For our first graph we created a scree plot. On the x axis we have the number of principal components (PC) and on the y axis we have the amount of variance explained per PC. Here we can see that the first few components can account for majority if not all the of variance explained in the data. And as the number of PCs increases to 8 through 10, there is little to no variance explained by those PCs.

```
# scree plot
    # the number of
pcaDF = pcaDF.append(pd.DataFrame({"expl_var": [0],
                                   "pc": [0],
                                   "cum_var": [0]}))

(ggplot(pcaDF, aes(x = "pc", y = "cum_var")) +
 geom_line() +
 geom_point() +
 theme_minimal() +
```

```
 ggtitle("Cumulative Variance Explained per Principal Component") +
 labs(x = "Number of Principal Components (PC)", y = "Cumulative Variance Explained") +
 ylim(0,1) +
 geom_hline(yintercept = 0.9))
```

We created another scree plot displaying the cumulative variance as we increase the number of PCs. Once again on the x axis we have the number of principal components (PC) and on the y axis we have the cumulative variance explained. As opposed to the previous graph that just explained the amount of cumulative variance for that one PC, here we can see the total amount of cumulative variance explained as the number of PC increases. Here we can see that with just one PC, it can account for more than 60% of the variance in the data. And in order for us to account for 90% of the original variance, we only need to keep 4 Principal Components.

Question 3: Is there a stronger linear relationship between the in state tuition vs early career pay or out of state tuition vs early career pay?

```
X_train, X_test, y_train, y_test = train_test_split(df["in_state_tuition"],
                                                    df["early_career_pay"],
                                                    test_size = 0.2)

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)

print("The MSE for the training set is:",
      mean_squared_error(y_train, model.predict(X_train.values.reshape(-1, 1))))
print("The R2 for the training set is:",
      r2_score(y_train, model.predict(X_train.values.reshape(-1, 1))))
print("The MSE for the testing set is:",
      mean_squared_error(y_test, model.predict(X_test.values.reshape(-1, 1))))
print("The R2 for the testing set is:",
      r2_score(y_test, model.predict(X_test.values.reshape(-1, 1))))



X_train, X_test, y_train, y_test = train_test_split(df["out_of_state_tuition"],
                                                    df["early_career_pay"],
                                                    test_size = 0.2)

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)

print("The MSE for the training set is:",
      mean_squared_error(y_train, model.predict(X_train.values.reshape(-1, 1))))
print("The R2 for the training set is:",
      r2_score(y_train, model.predict(X_train.values.reshape(-1, 1))))
```

```
print("The MSE for the testing set is:",
      mean_squared_error(y_test, model.predict(X_test.values.reshape(-1, 1))))
print("The R2 for the testing set is:",
      r2_score(y_test, model.predict(X_test.values.reshape(-1, 1))))
```

We used the linear model explained in the analysis plan. Once again we performed model validation to make sure our model does well on data that it's never seen before and doesn't pick up on any "patterns" from our sample data that doesn't apply to our entire data set. We didn't have to z-score our variables since they are both on the same scale (in USD). After performing a linear regression along with model validation, we compared the R^2 function to see which variable can account for more variation in the data set. Here we were able to see that the second model was able to account for more the variance in the data set. Thus, we can conclude that there is a slightly stronger linear relationship between out-of-state tuition and early career pay than in-state tuition and early career pay.

For the first model, the R^2 is very low (0.28140 for the training data set and 0.27871 for the testing data set). A R^2 if 1 means that 100% of variability in early career pay can be accounted for by our predictor variable. One good thing from the first model is that the MSE is very small (0.72724 for the training data set and 0.68662 for the testing data set). MSE stands for the mean squared error. Ideally we want a MSE of 0 because that means our model was a prefect prediction for all of our variables and our model had no error. After looking and comparing at both loss function, we can conclude that the first model is slightly overfit because the MSE and R^2 is slightly higher for the training data than the testing data set.

For the second model, the R^2 is slightly higher than the first model (0.43647 for the training data set and 0.45511 for the testing data set). However, in terms of a general case, the R^2 is still very low, about 43 to 45%. The MSE is smaller than the MSE for the first model which makes sense because the R^2 is slightly higher, which means it is able to account for more variation in the data set. The MSE for the training set is 0.53757 and 0.64469 for the testing data set. Once again the MSE are fairly close to 0 which is a good thing, because 0 means that our model was perfect, with no error. Once again, after looking and comparing both loss functions, we can conclude that the model is neither overfit nor underfit because the MSE and R^2 are very close to one another for the training and testing data.

```
# scatter plot
    # in-state tuition in USD VS early career pay in USD
(ggplot(df, aes(x = "in_state_tuition", y = "early_career_pay")) +
 geom_point() +
 geom_smooth(method = "lm", color = "red") +
 theme_minimal() +
```

```
 ggtitle("In State Tuition VS Early Career Pay") +
 labs(x = "In State Tuition (USD)", y = "Early Career Pay (USD)"))
```

For our first plot, we created a scatterplot comparing the In State Tuition VS Early Career Pay. Here on the x axis we have the amount of people paid for in state tuition in USD and on the y axis we have the amount people are paid for the start of their career in USD. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low because majority of the data points do not fall close to the line. The data points are pretty spread out and not close to each other or the red line. We can also see that as in-state tuition increases, we can see that early career pay also increases at a pretty slow rate.

```
(ggplot(df, aes(x = "out_of_state_tuition", y = "early_career_pay")) +
 geom_point() +
 geom_smooth(method = "lm", color = "red") +
 theme_minimal() +
 ggtitle("Out of State Tuition VS Early Career Pay") +
 labs(x = "Out of State Tuition (USD)", y = "Early Career Pay (USD)"))
```

For our second plot, we created another scatterplot comparing the Out of State Tuition VS Early Career Pay. Here on the x axis we have the amount people paid for out of state tuition in USD and on the y axis we have the amount people are paid for the start of their career in USD. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low, but still higher than our first scatterplot (in state tuition VS early career pay) because majority of the data points do not fall close to the line. More data points lie closer and are more evenly spread out to the line of best fit on this graph than the one above. We can also see that as in state tuition increases, we can see that early career pay also increases at a steeper rate than in state tuition vs early career pay.

Nicole Matsumoto

Question 1: Does race play a significant factor in student enrollment in school? (Many schools want diversity, but do they actually reach perfect diversity or still have a significant amount of a certain race?)?

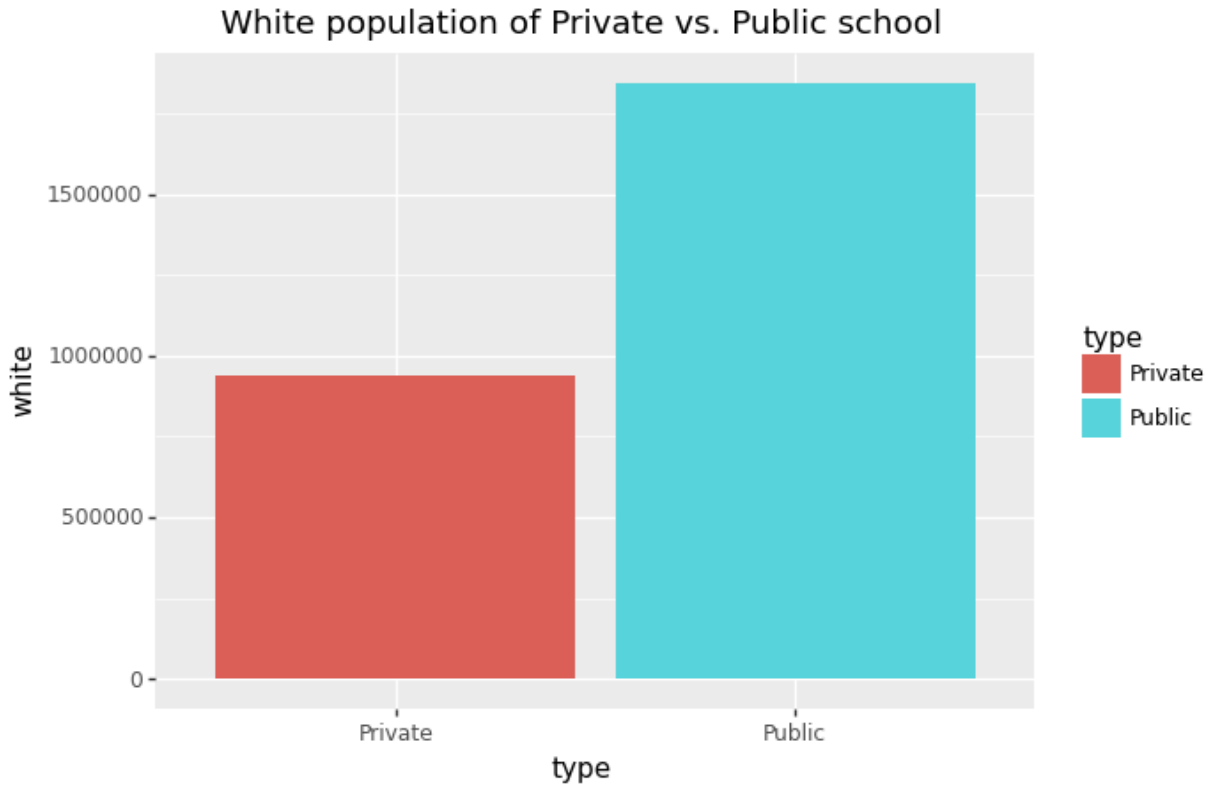
```
df['asian']=df['asian'].astype(str).str.replace(',', '').astype(int)
df['white']=df['white'].astype(str).str.replace(',', '').astype(int)
df['black']=df['black'].astype(str).str.replace(',', '').astype(int)
df['hispanic']=df['hispanic'].astype(str).str.replace(',', '').astype(int)
df['women']=df['women'].astype(str).str.replace(',', '').astype(int)
df['enrollment']=df['enrollment'].astype(str).str.replace(',', '').astype(int)
```

```
df3 = df.groupby(['type'], as_index=False).sum()
df3.head()

df3 = df3.drop(labels=0, axis=0)

race = df3[['type','asian','black','white', 'hispanic', 'women', 'enrollment', 'in_sta
race.head()
```

	type	asian	black	white	hispanic	women	enrollment	in_state_tot
1	Private	115754	156672	936811	131477	936391	1704277	197371
2	Public	129483	347005	1841531	264528	1628242	2970971	44235

```
white_pop = df3["white"]/ df3['enrollment']
print(white_pop)
(ggplot(race, aes(x = 'type', y = 'white', fill = 'type')) + geom_bar(stat= "identity'
ggtitle("White population of Private vs. Public school"))
```

```
1    0.549682
2    0.619841
dtype: float64
```

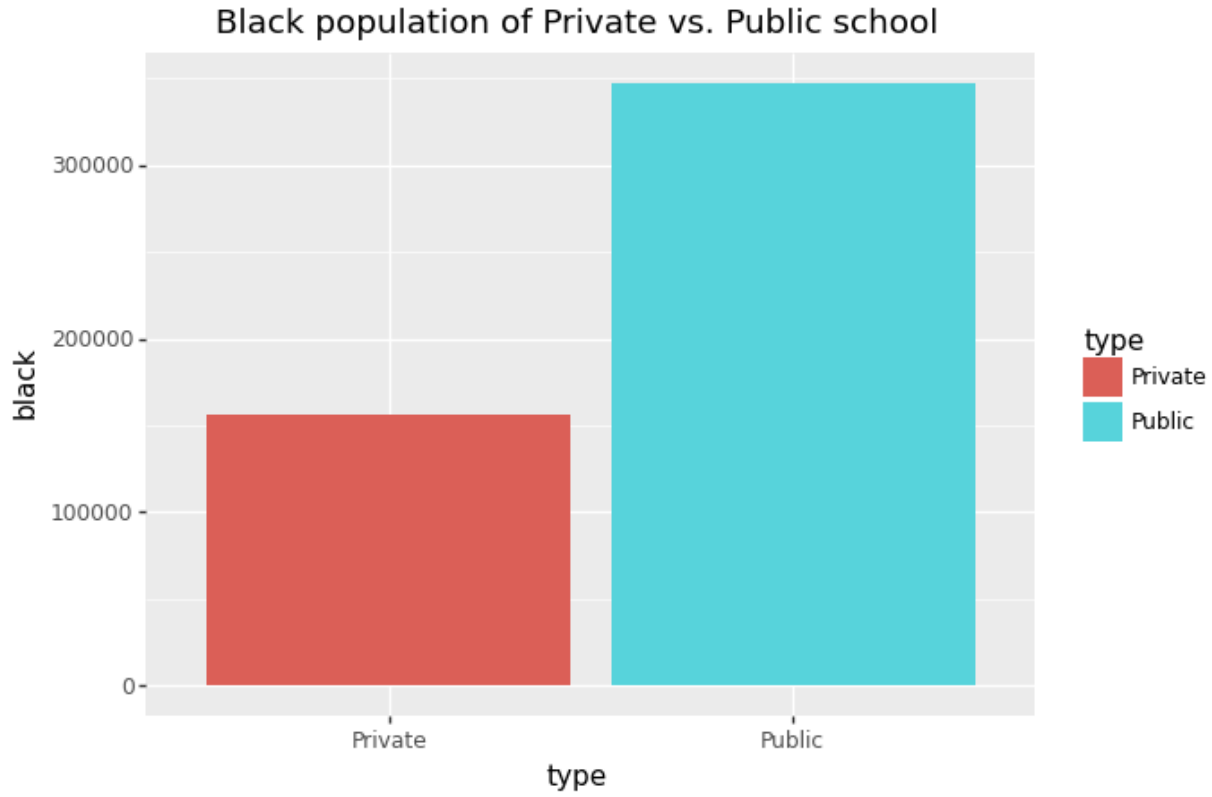



```
<ggplot: (8736295112297)>
```

For the first plot, it shows the enrollment of the white population in private schools and public schools. The overall percentage of the white population in private school is about 55%. The overall percentage of the white population in public school is about 62%. Private and public school percentage about the same and are pretty high. This graph proves the theory that there is not perfect diversity and still has a significant amount of white people vs. other races

```
black_pop = df3["black"]/ df3['enrollment']
print(black_pop)
(ggplot(race, aes(x = 'type', y = 'black', fill = 'type')) + geom_bar(stat= "identity'
ggtitle("Black population of Private vs. Public school"))
```

```
1    0.091929
2    0.116799
dtype: float64
```



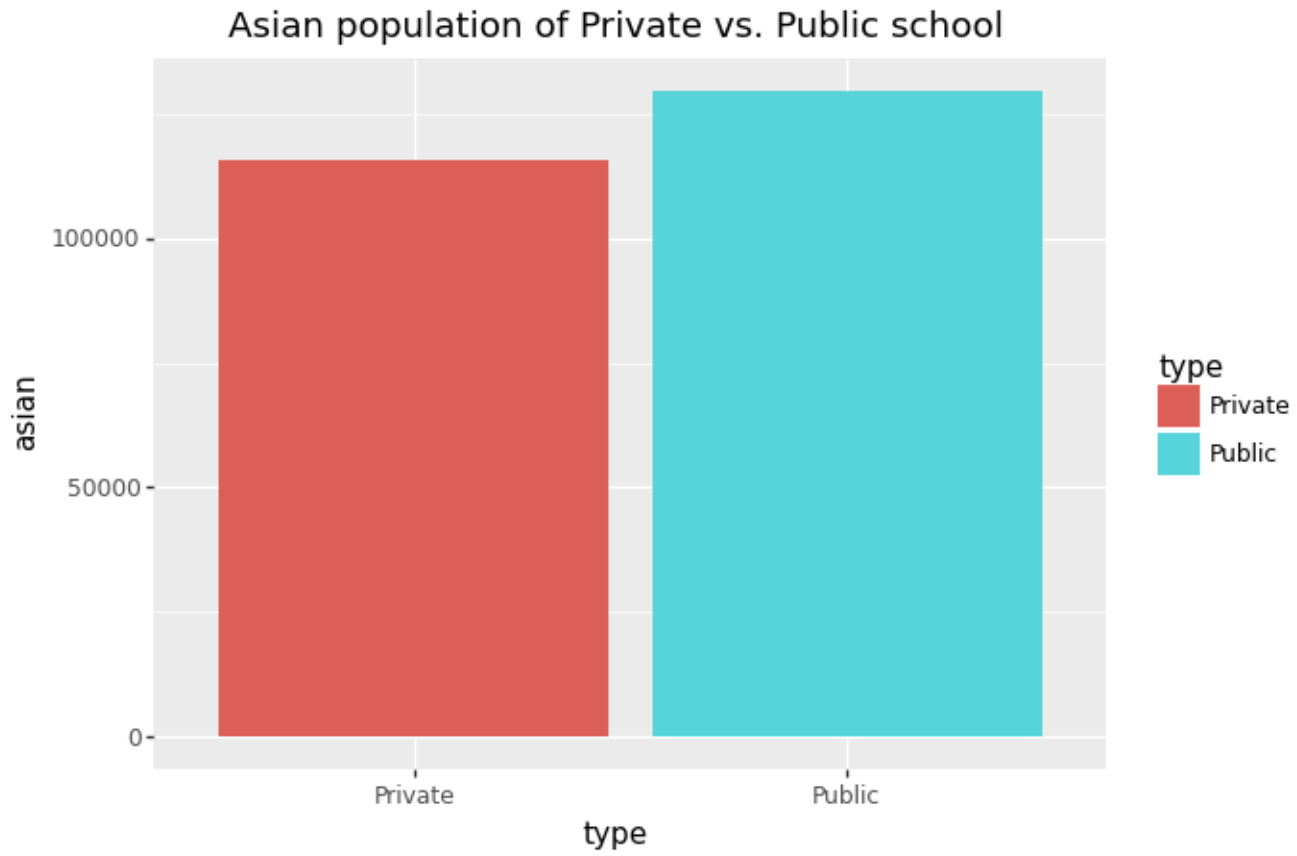

```
<ggplot: (8736295112325)>
```

For the second plot, it shows the enrollment of the black population in private schools and public schools. The overall percentage of the black population in private school is about 9%. The overall percentage of the black population in public school is about 12%. Private and public school percentage about the same but are both less than 15%. They have the second highest percentage but is still significantly lower than the white population enrolled in college.

```
asian_pop = df3["asian"]/ df3['enrollment']
print(asian_pop)
```

```
(ggplot(race, aes(x = 'type', y = 'asian', fill = 'type')) + geom_bar(stat= "identity'
ggtitle("Asian population of Private vs. Public school"))
```

```
1    0.067920
2    0.043583
dtype: float64
```

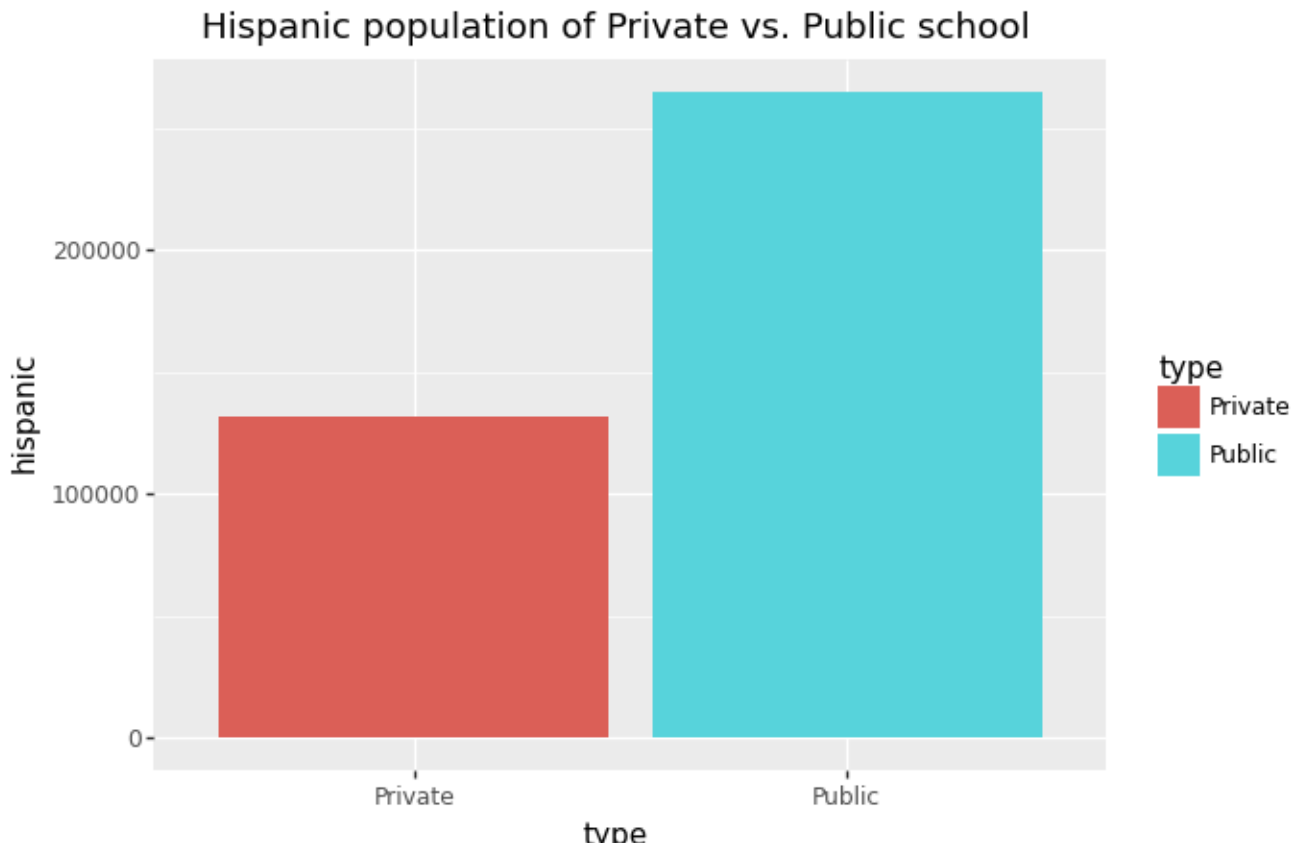



```
<ggplot: (8736295111321)>
```

For the third plot, it shows the enrollment of the asian population in private schools and public schools. The overall percentage of the asian population in private school is about 7%. The overall percentage of the asian population in public school is about 4%. Private and public school percentage about the same and are under 10%. This graph is interesting because the percentage of asians in private schools are higher than asians in public schools which is different than the other three.

```
hispanic_pop = df3["hispanic"]/ df3['enrollment']
print(hispanic_pop)
(ggplot(race, aes(x = 'type', y = 'hispanic', fill = 'type')) + geom_bar(stat= "identi
ggtitle("Hispanic population of Private vs. Public school"))
```

```
1    0.077145
2    0.089038
dtype: float64
```



For the fourth plot, it shows the enrollment of the hispanic population in private schools and public schools. The overall percentage of the hispanic population in private school is about 8%. The overall percentage of the hispanic population in public school is about 9%. Private and public school percentage about the same and are less than 10%. The percentages for this plot is the closest together showing that they are the most consistant through public and private schools.

Analysis

b) I cleaned the data even further to make all the races integers instead of objects. Then, I made bar graphs showing the different races(black, white, asian, hispanic) vs. the type of school (public or private). The private bar is red and the public bar is blue to easily show the difference between public and private schools. I also found the percent by dividing the different races by enrollment of either public or private.

This answer is important because it shows that there is not equal diversity and that the white population makes up over 50% of all enrollment. Each of all the other races are under 15%. The lowest percentage is 4.36% which is asians in public schools. This can be helpful to colleges because it shows the need for more diversity in schools and exposes which races they accept more than others and that it is not even close to equal (20/20/20/20)

Question 2: Check if the percentage of minorities of each school are over a certain percentage based on specific predictors?

Double-click (or enter) to edit

```
df["minority_percentage"].mean()
```

```
0.26691662665967036
```

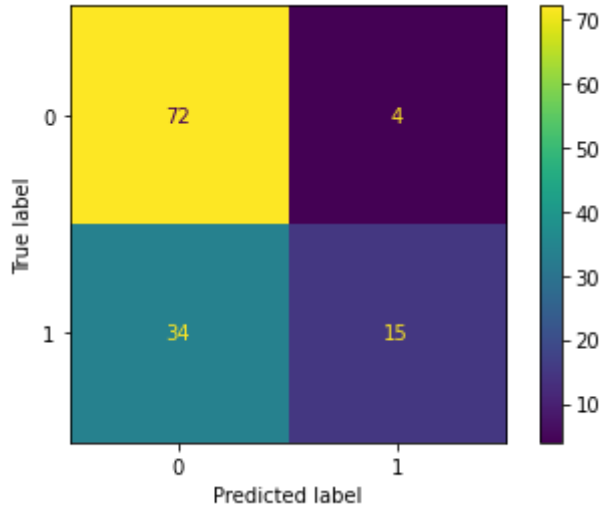
```
df["minority_percentage_over_mean"] = np.where((df["minority_percentage"] > 0.27),1,0)
df["Public"] = np.where((df["type"] == "Public"),1,0)
df["Private"] = np.where((df["type"] == "Private"),1,0)
```

```
predictors = ["West","South","Midwest","Northeast","Private","Public"]
X = df[predictors]
y = df["minority_percentage_over_mean"]

X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2, random_state

tree_model = DecisionTreeClassifier(random_state = 1234, max_depth = 3)

tree_model.fit(X_train,y_train)
```

```
DecisionTreeClassifier(max_depth=3, random_state=1234)
```

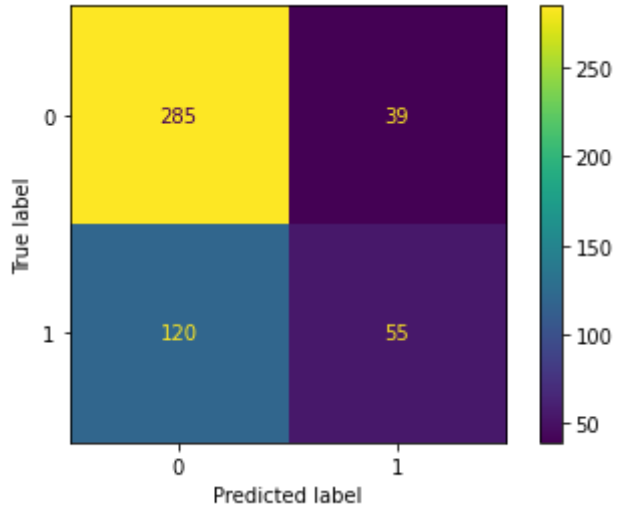
```
plot_confusion_matrix(tree_model, X_test, y_test)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f213c536bd0
```




```
plot_confusion_matrix(tree_model, X_train, y_train)
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x7f213ca28110>
```

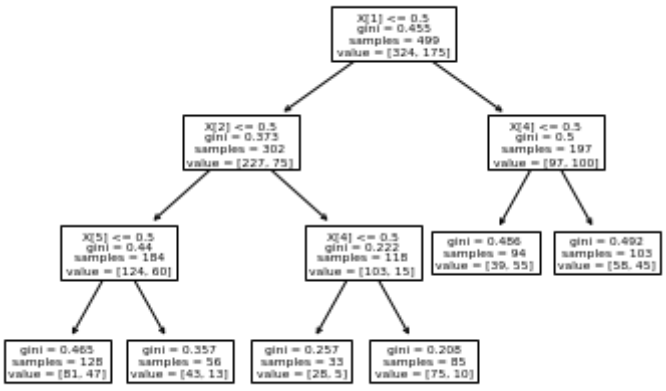



```
tree_model.get_depth()
```

```
3
```

```
tree.plot_tree(tree_model)
```

```
[Text(197.83636363636364, 190.26, 'X[1] <= 0.5\ngini = 0.455\nsamples = 499\nvalu
 Text(121.74545454545455, 135.9, 'X[2] <= 0.5\ngini = 0.373\nsamples = 302\nvalue
 Text(60.872727272727275, 81.53999999999999, 'X[5] <= 0.5\ngini = 0.44\nsamples =
 Text(30.436363636363637, 27.180000000000007, 'gini = 0.465\nsamples = 128\nvalue
 Text(91.30909090909091, 27.180000000000007, 'gini = 0.357\nsamples = 56\nvalue =
 Text(182.61818181818182, 81.53999999999999, 'X[4] <= 0.5\ngini = 0.222\nsamples
 Text(152.1818181818182, 27.180000000000007, 'gini = 0.257\nsamples = 33\nvalue =
 Text(213.05454545454546, 27.180000000000007, 'gini = 0.208\nsamples = 85\nvalue
 Text(273.92727272727274, 135.9, 'X[4] <= 0.5\ngini = 0.5\nsamples = 197\nvalue =
 Text(243.4909090909091, 81.53999999999999, 'gini = 0.486\nsamples = 94\nvalue =
 Text(304.3636363636364, 81.53999999999999, 'gini = 0.492\nsamples = 103\nvalue =
```



For our first plot we created a decision tree and used a max depth of 3 because 5 was too small and was unreadable. The most impactful factor is X[1] which is Midwest. The gini impurity measures the

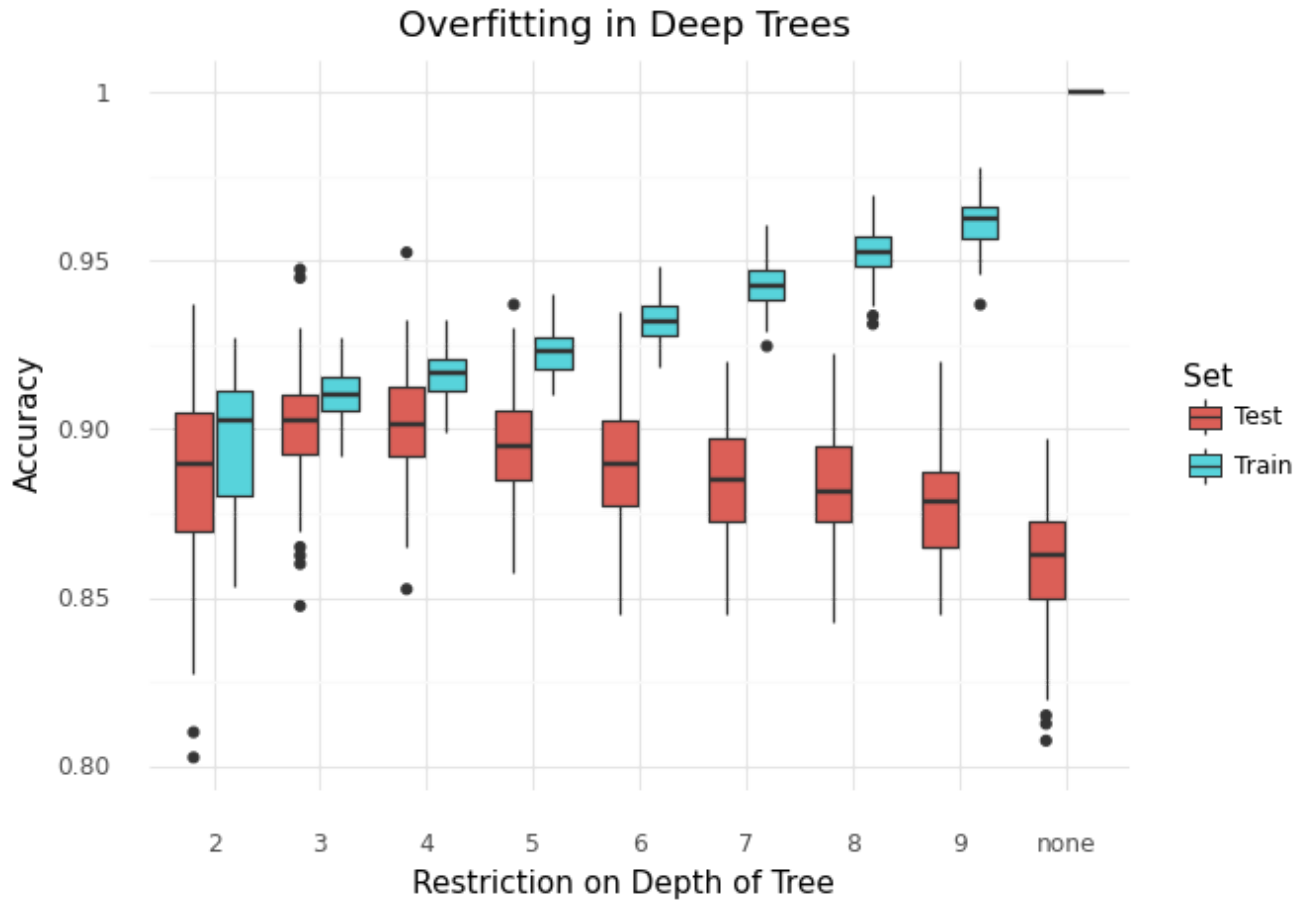
degree or probability of a particular variable being wrongly classified when it is randomly chosen. The gini impurity score is .455 and is sampled around 409. Region plays a big factor in deciding.

```
def simTree(n = 200, max_depth = "none"):

    n2 = int(n/2)

    #---generate data-----------------------------------
    # generate predictors, only x1 and x2 are related to the outcome
    x1 = np.concatenate([np.random.normal(0,1,n2),
                         np.random.normal(2,1,n2)])
    x2 = np.random.normal(0,1,n)
    x3 = np.concatenate([np.random.binomial(1,0.2,n2),
                        np.random.binomial(1,0.9, n2)])

    # generate outcome
    outcome = np.repeat(["A","B"], n2)

    # data frame
    df = pd.DataFrame({"x1": x1, "x2" : x2, "x3": x3, "out": outcome})

    #---set X and y-------------------------------------
    X = df[["x1","x2","x3"]]
    y = df[["out"]]
    X_train, X_test, y_train, y_test = train_test_split(X,y, test_size = 0.2)


    #---build models------------------------------------

    # if there is no max depth, build regular tree
    if max_depth == "none":
        tree = DecisionTreeClassifier()

    # otherwise, set max_depth
    else:
        tree = DecisionTreeClassifier(max_depth = max_depth)

    tree.fit(X_train,y_train)


    #---get training and test Accuracy---------------------
    train_acc = accuracy_score(y_train,tree.predict(X_train))
    test_acc = accuracy_score(y_test,tree.predict(X_test))

    return(pd.DataFrame({"Set": ["Train", "Test",],
                         "Depth": [str(max_depth), str(max_depth)],
                         "Acc": [train_acc,
                                 test_acc]}))
```

```
sims = [simTree(n = 2000, max_depth = x) for x in np.repeat(range(2,10), 100)]
sims = sims + [simTree(n = 2000, max_depth = "none") for i in range(0,100)]
sims_df = pd.concat(sims)

sims_df.head()
```

	Set	Depth	Acc
0	Train	2	0.853125
1	Test	2	0.867500
0	Train	2	0.881250
1	Test	2	0.897500
0	Train	2	0.907500

```
(ggplot(sims_df, aes(x = "Depth", y = "Acc", fill = "Set")) +
geom_boxplot() + theme_minimal() +
labs(x = "Restriction on Depth of Tree",
    y = "Accuracy",
    title = "Overfitting in Deep Trees"))
```




```
<ggplot: (8736294901385)>
```

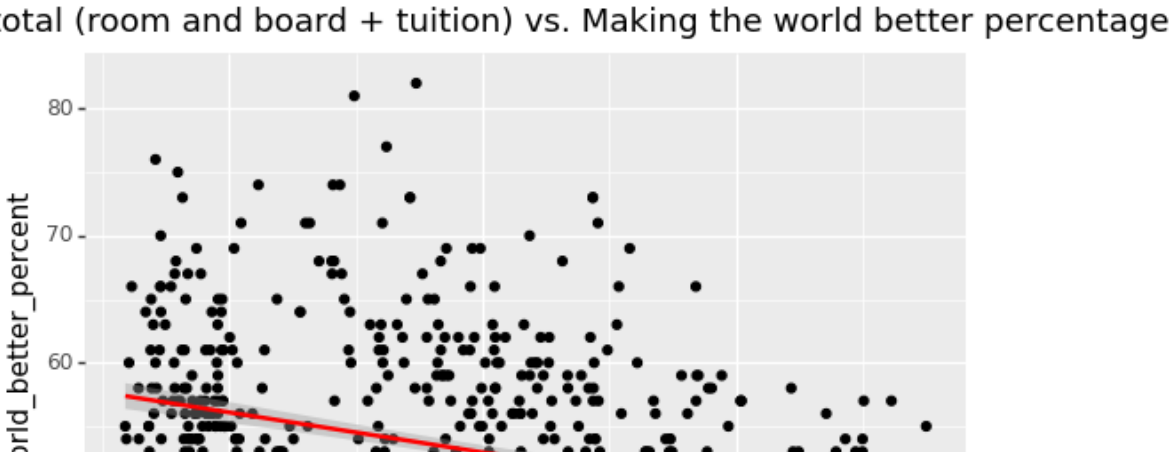
For our second plot we created a box plot comparing the depth and accuracy of the overfitting deep trees. As you put more trees, the model becomes more overfit because your using more predictors making it more complex. The test and train split around 4 showing that we should use around 3 to 5 trees because we do not want our model to be overfit.

Analysis

b) A decision tree model is a non-parametric supervised learning method used for classification and regression. THe goal is to predict the value of a target variable by simple decision rules infereed from predictors. The first thing we have to do is find the mean because we are using the decision tree to check whether the percentage of minorities at each school excided a certain percentage based on the predictors. I used the different regions (West, South, Midwest, and Northwest) and type (public and private). In our model, we used max depth of 3 because a max depth of 5 (which is what is predicted), I was unable to read. I also did a train test split and then fir the model. Then I created a confusion matrix, one for test and one for train. For the test, it was able to predict most of the time the correct value but 34 times it predicted no when it was actualy yes and 72 times it predicted no when it was no. For the test, it was pretty accurate too because it predicted no when it was no 285 times and 120 timrs it predicted no when it was yes.

Question 3: Is there a strong linear relationship between total (room and board and tuition) and career pay (early and mid) vs. making the world a better place? (one model for in_state_total and out of state total for each race)

```
(ggplot(df, aes(x = "in_state_total", y = "make_world_better_percent")) + geom_point()
 geom_smooth(method = "lm", color = "red") +
ggtitle("In-state total (room and board + tuition) vs. Making the world better percent
```



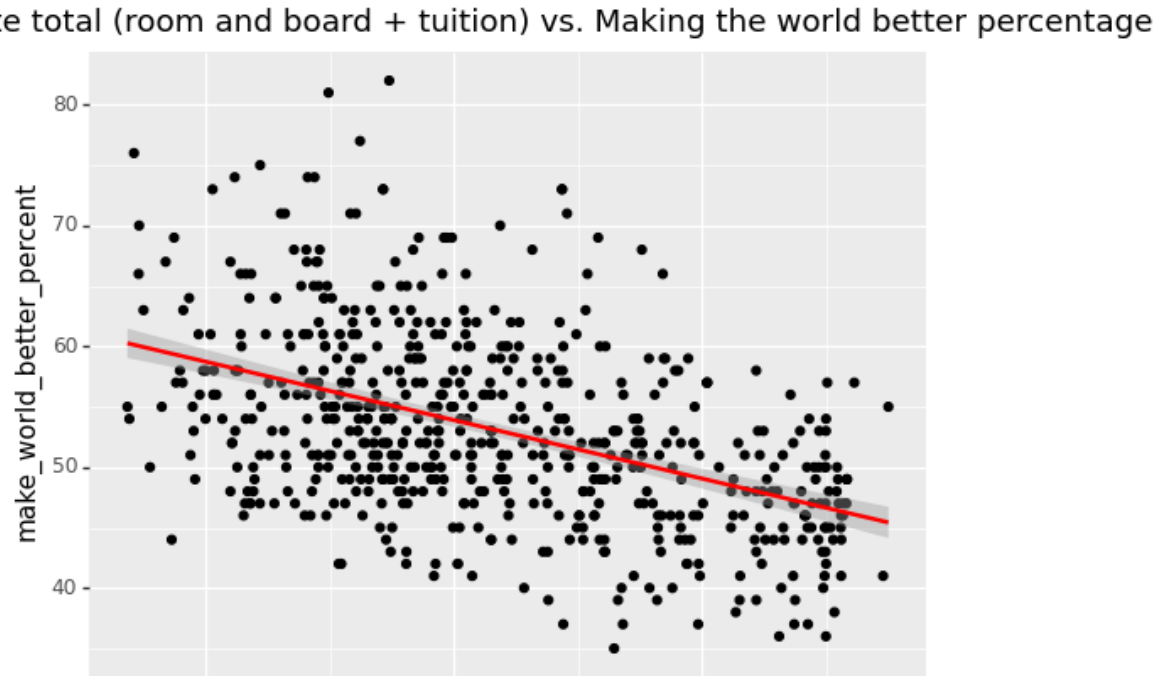
For our first plot, we created a scatterplot comparing the In State total cost (room and board + tuition) VS make the world better percent. Here on the x axis we have the amount of people paid for in state total in USD and on the y axis we have the percent of people who make the world better. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low because majority of the data points do not fall close to the line. The data points are pretty spread out and not close to each other. There is not a really correlation in this plot.

```
<ggplot: (8736204846841)>
```

```
X_train, X_test, y_train, y_test = train_test_split(df["in_state_total"], df["make_wor

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)

print("In state total vs. Make World Better Percent")
print("Mean squared error for train model:", mean_squared_error(y_train, model.predict
print("R2 score for train model:", r2_score(y_train, model.predict(X_train.values.resh
print("Mean squared error  for test model:", mean_squared_error(y_test, model.predict(
print("R2 score for test model:", r2_score(y_test, model.predict(X_test.values.reshape
```

```
In state total vs. Make World Better Percent
Mean squared error for train model: 52.52655969606042
R2 score for train model: 0.12861701028892192
Mean squared error  for test model: 51.649472457460185
R2 score for test model: 0.1679317381711356
```

```
(ggplot(df, aes(x = "out_of_state_total", y = "make_world_better_percent")) + geom_poi
 geom_smooth(method = "lm", color = "red") +
ggtitle("Out-of-state total (room and board + tuition) vs. Making the world better per
```

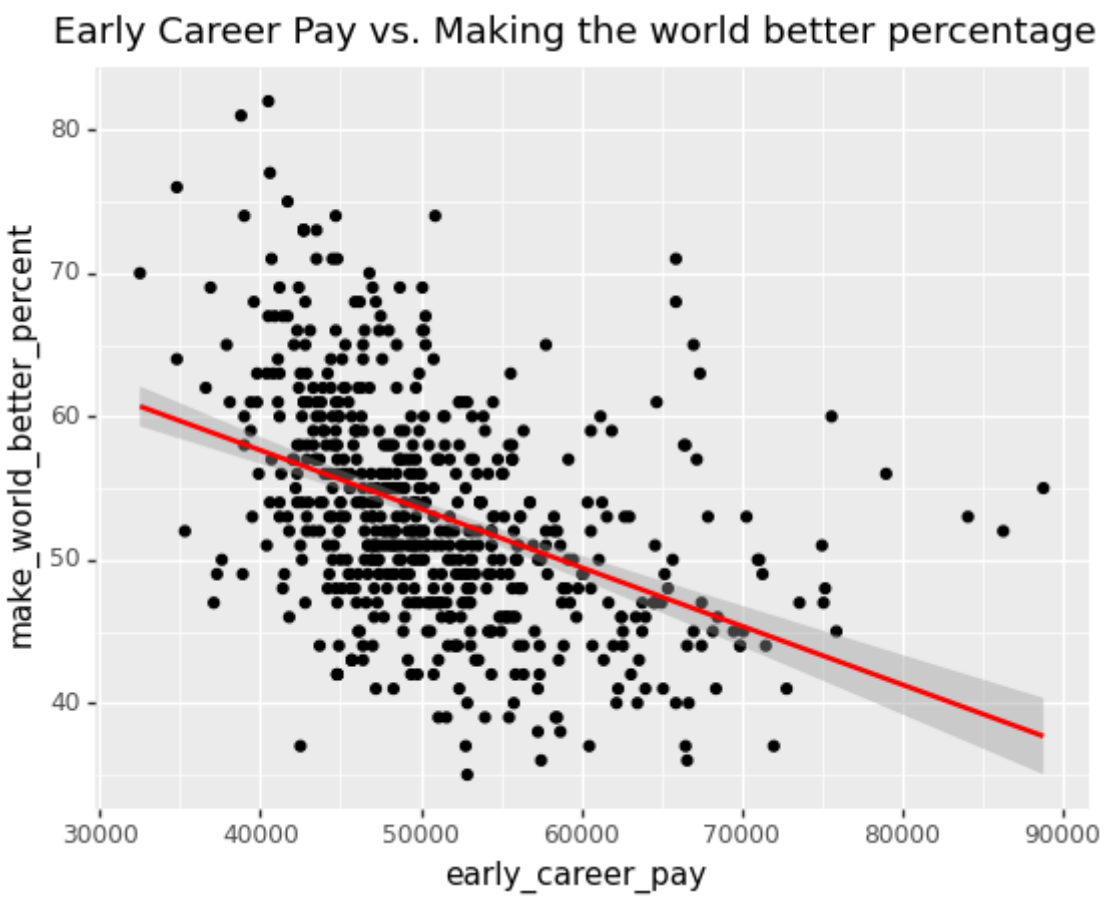


For our second plot, we created a scatterplot comparing the Out of state total cost (room and board + tuition) VS make the world better percent. Here on the x axis we have the amount of people paid for out of state total in USD and on the y axis we have the percent of people who make the world better. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low because majority of the data points do not fall close to the line. We can also see that this plot is better because the points are a little better on the line then the first plot. The data points are pretty spread out and not close to each other. There is not a really correlation in this plot.

```
X_train, X_test, y_train, y_test = train_test_split(df["out_of_state_total"], df["make

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)

print("Out of State total vs. Make World a better place percentage")
print("Mean squared error for train model:", mean_squared_error(y_train, model.predict
print("R2 score for train model:", r2_score(y_train, model.predict(X_train.values.resh
print("Mean squared error  for test model:", mean_squared_error(y_test, model.predict(
print("R2 score for test model:", r2_score(y_test, model.predict(X_test.values.reshape
```

```
Out of State total vs. Make World a better place percentage
Mean squared error for train model: 46.09208265800356
R2 score for train model: 0.20903087398900555
Mean squared error  for test model: 54.768636152865206
R2 score for test model: 0.20898466525995385
```

```
(ggplot(df, aes(x = "early_career_pay", y = "make_world_better_percent")) + geom_point
 geom_smooth(method = "lm", color = "red") +
ggtitle("Early Career Pay vs. Making the world better percentage"))
```




```
<ggplot: (8736294492057)>
```

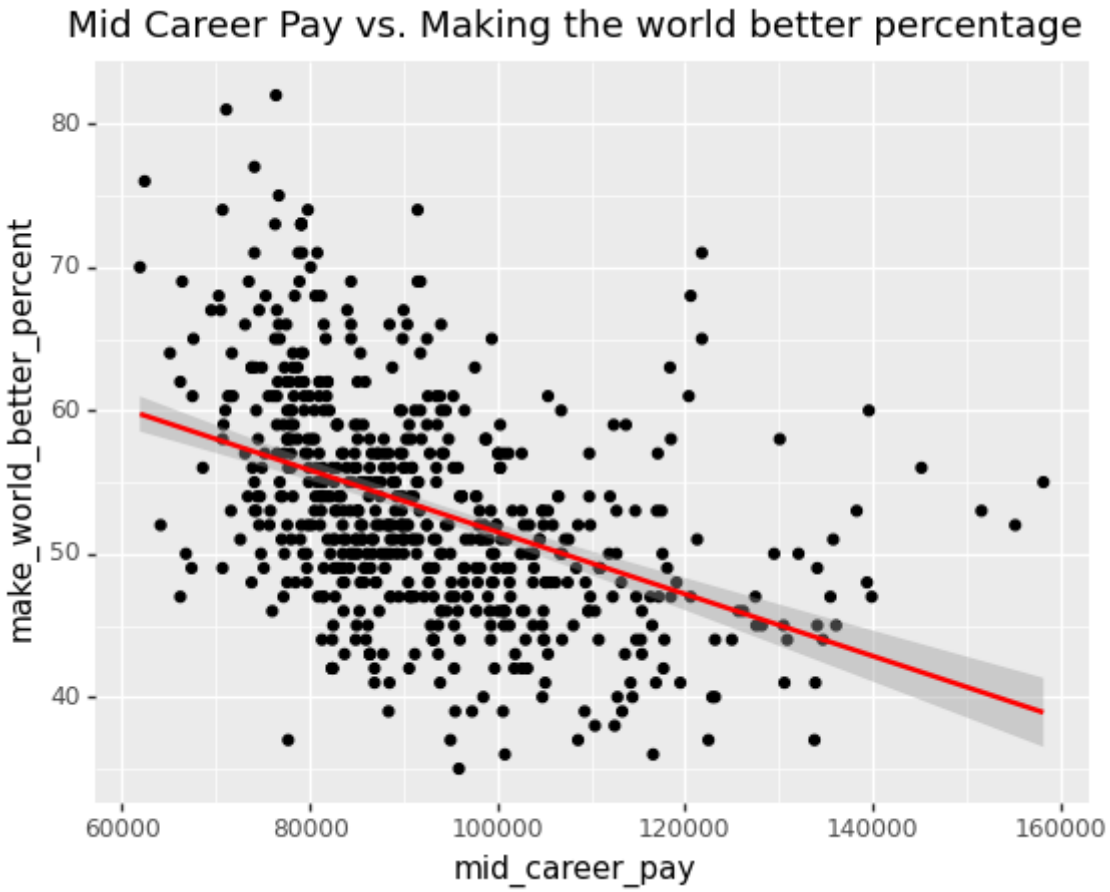
For our third plot, we created a scatterplot comparing the early career pay VS make the world better percent. Here on the x axis we have the amount of people make in their early career in USD and on the y axis we have the percent of people who make the world better. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low because majority of the data points do not fall close to the line. The data points are close togerther on the left to middle and are spreaded out around. There is a decent negative correlation.

```
X_train, X_test, y_train, y_test = train_test_split(df["early_career_pay"], df["make_w

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)

print("Early Career Pay vs. Make World a better place percentage")
print("Mean squared error for train model:", mean_squared_error(y_train, model.predict
print("R2 score for train model:", r2_score(y_train, model.predict(X_train.values.resh
print("Mean squared error  for test model:", mean_squared_error(y_test, model.predict(
print("R2 score for test model:", r2_score(y_test, model.predict(X_test.values.reshape
```

```
Early Career Pay vs. Make World a better place percentage
Mean squared error for train model: 50.67065214146012
R2 score for train model: 0.17247297485304292
```

```
Mean squared error  for test model: 46.36425025440325
R2 score for test model: 0.20157140565100518
```

```
(ggplot(df, aes(x = "mid_career_pay", y = "make_world_better_percent")) + geom_point()
 geom_smooth(method = "lm", color = "red") +
ggtitle("Mid Career Pay vs. Making the world better percentage"))
```




```
<ggplot: (8736294461917)>
```

For our last plot, we created a scatterplot comparing the mid career pay VS make the world better percent. Here on the x axis we have the amount of people make in their mid career in USD and on the y axis we have the percent of people who make the world better. We also have a red line that shows us the linear relationship between the two variables. We can see why our $R2$ was so low because majority of the data points do not fall close to the line. The data points are close togerther on the left to middle and are spreaded out around. This looks pretty similar to early career but has a worst R2 than early career. There is a decent negative correlation.

```
X_train, X_test, y_train, y_test = train_test_split(df["mid_career_pay"], df["make_wo

model = LinearRegression()
model.fit(X_train.values.reshape(-1, 1), y_train)
```

```
print("Mid Career Pay vs. Make World a better place percentage")
print("Mean squared error for train model:", mean_squared_error(y_train, model.predict
print("R2 score for train model:", r2_score(y_train, model.predict(X_train.values.resh
print("Mean squared error  for test model:", mean_squared_error(y_test, model.predict(
print("R2 score for test model:", r2_score(y_test, model.predict(X_test.values.reshape
```

```
Mid Career Pay vs. Make World a better place percentage
Mean squared error for train model: 49.25132918824204
R2 score for train model: 0.1932100354434192
Mean squared error  for test model: 50.05978507306763
R2 score for test model: 0.1444021394026207
```

Analysis

b) We cleaned the data to make sure that it was clean and accurate. Then I performed a linear regression along with a validation, where we compared the R2 and the MSE for the train and the test. I also created 4 different plots, 2 for total which is in-state and out-of-state total (room and board + tuition) vs. make the world better percentage and 2 for career pay which is early and mid career vs. make the world better percentage to how the plots looked and if there was a relationship. We can conclude from both of these plots that there is not a strong relationship. We can also compare total cost and career pay and conclude it has a slightly higher relationship for career pay than total costs.

For the first model (in-state total vs. make the world better percent), the R2 is very low. The R2 score for train model is 14.62% and the R2 score for the test model is 8.65%. R2 for the train is slightly higher than the test for R2 showing that this model is slightly overfit. For the second model (out-of-state total vs. make the world better percent), the R2 is also low but way higher than the first model. The R2 for the train is 21.39% and the R2 for the test is 20.46% which is pretty much the same but like the first model, it is barely overfit. Both of these models show that the independrnt variable is not explaining much in the variation of the dependent variable. For the third model (early career vs. make the world better percent), the R2 for the train is 16% an the R2 for the test is 25.52% which means that it is underfit because the test is greater than the train. For the fourth model (mid career vs. make the world better percent), the R2 for the train is 18.28% and the R2 for the test is 19.62% showing thay they are pretty much the same, but it is slightly overfit because the test is barely grreater than the train.

MSE stands for the mean squared error, which we want a MSE of 0 because that means our model is a perfect prediction for all the variables. This is important even if you are a better person, it has little to do with the total cost for in-state or out-of-state. In addition, the early career and mid career also has has ;ittle to do with a person making the world better. This could be helpful because maybe colleges want to take a closer look on giving people's scholarships if they have a greater positive impact on the world.

```
!wget -nc https://raw.githubusercontent.com/brpy/colab-pdf/master/colab_pdf.py
from colab_pdf import colab_pdf
colab_pdf('cpsc392Kai.ipynb')
```