Mini Project 2, Daniel Avila

```
import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve, roc_auc_score

os.chdir("C:/Users/AVILA/OneDrive/Documents/GitHub/Machine-Learning--Harris/Mini-Project-2")

df = pd.read_csv("Data-Audit.csv")
```

2.

(a) The 3 variables that I could see having an impact on the true model from our dataset would be the sector the firm is in, PARA A (which is the discrepancies found in planned expenditure), PARA B (which is the discrepancies found in unplanned expenditures), and Money Value (which is the firm revenue in the past 2 years). Since I dont know how the Inherent Risk or the Score variables are produced, it is hard for me to argue that those would have an impact on the true model, however some level of risk measurement is valuable to predictive models in these contexts. I believe these features are good guesses at the true underlying relationship because 1. Para A contains discrepancies that you would expect, which is self reported and could be manipulated, 2. Para B is also self reported and unexpected expenses are by definition variable, which means that there exists the opportunity to "fudge" these numbers, and 3. the firm income plays a big role because significant changes in either the positive or negative direction for firm income could incentivize a firm to engage in tax evasion, i.e. large positive income changes would incentivize a firm to try and keep more of their revenue, while large negative income changes would incentivize a firm to try and increase the amount of "losses" which would reduce their tax burden.

If we are considering other variables outside of this dataset, we might be interested in whether there was a new CEO (who would be incentivized to show company performance improvements, which could be done through financial manipulation), an unusually large number of "consulting" or "legal" contracts...etc.

(b) The LPM would have biased estimates without the interaction term, because the interaction term would provide the "boosted" additional effect of both the individual variables and the interaction. When the interaction term is missing, that would indicate an understimate of the true effect.

KNN would have an advantage in this senario bacause KNN is not dependent on these interaction terms. KNN is interested in calculating the centroid of a series of data points relative to the distance between other datapoints to produce groupings. Therefore, if the firms that are cheating on their tax returns are displaying similar charactaristics through the data, then we would see cheating firms grouped together in a way that KNNN would be able to catch.

3.

(a)

```
print(f"df shape before dropping na's: {df.shape}")
df = df.dropna()
print(f"df shape after dropping na's: {df.shape}")

X_train, X_test, y_train, y_test = train_test_split(df.loc[:, "Sector_score":"Audit_Risk"], c

model = LinearRegression()
model.fit(X_train, y_train)
print()
print(f"model score: {model.score(X_test, y_test)}")

y_pred = model.predict(X_test)
y_pred_5 = np.where(y_pred > .5 , 1, 0)
y_pred_6 = np.where(y_pred > .6 , 1, 0)

print()
print(f"confusion matrix, .5: ")
print(f"{confusion_matrix(y_test, y_pred_5)}")
tn5, fp5, fn5, tp5 = confusion_matrix(y_test, y_pred_5).ravel()
```

```
df shape before dropping na's: (776, 11)
df shape after dropping na's: (775, 11)

model score: 0.4611529930123326

confusion matrix, .5:
[[221   8]
 [ 29 130]]
```

(b)

```
print(f"confusion matrix, .6: ")
print(f"{confusion_matrix(y_test, y_pred_6)}")
tn6, fp6, fn6, tp6 = confusion_matrix(y_test, y_pred_6).ravel()
```

```
confusion matrix, .6:
[[225   4]
 [ 39 120]]
```

(c)

```
error_rate_5 = (fp5 + fn5) / (tn5 + tp5 + fn5 + fp5)
print(f"error rate for .5 threshold: {round(error_rate_5, 4)*100}%")

error_rate_6 = (fp6 + fn6) / (tn6 + tp6 + fn6 + fp6)
print(f"error rate for .6 threshold: {round(error_rate_6, 4)*100}%")
```

```
error rate for .5 threshold: 9.54%
error rate for .6 threshold: 11.08%
```

We find that the .5 threshold has a lower error rate than the .6 threshold, therefore we would expect the .5 threshold to provide more accurate predictions.

(d)

```
evasion5 = tp5 / (fp5 + tp5)
print(f"{round(evasion5, 2) * 100} % of predicted firms actually evaded their taxes, .5 thres

evasion6 = tp6 / (fp6 + tp6)
print(f"{round(evasion6, 2) * 100} % of predicted firms actually evaded their taxes, .6 thres
```

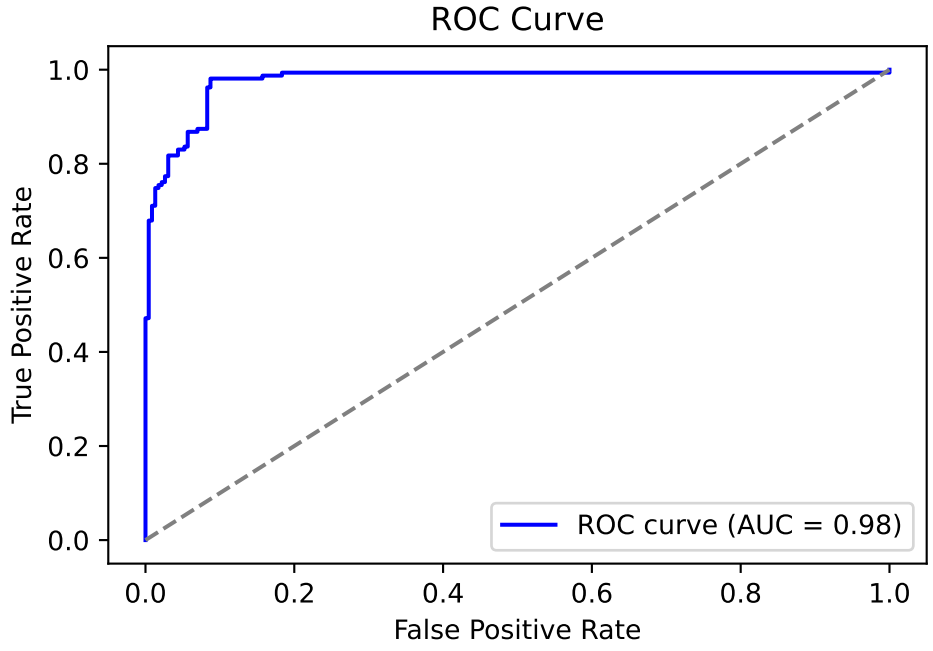
```
94.0 % of predicted firms actually evaded their taxes, .5 threshold
97.0 % of predicted firms actually evaded their taxes, .6 threshold.
```

The proportion of the firms that were predicted to evade their taxes and actually did evade their taxes would be the inverse of the error rate. Therefore, for the .5 threshold, we have ~90% of predicted firms actually evading their taxes. For the .6 threshold, we have ~89% of predicted firms actually evading their taxes.

(e)

```
fpr, tpr, thresholds = roc_curve(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred)
print(f"ROC AUC Score: {roc_auc}")
plt.plot(fpr, tpr, color="blue", label = f"ROC curve (AUC = {roc_auc:.2f})")
plt.plot([0,1], [0,1], color = 'gray', linestyle = "--")
plt.legend(loc = "lower right")
plt.title("ROC Curve")
plt.xlabel("False Positive Rate")
plt.ylabel("True Positive Rate")
plt.show()
```

```
ROC AUC Score: 0.9759138721814837
```



4.

In this context, false negatives mean that a firm was labeled as not committing tax fraud, but actuallly was commiting tax fraud. Therefore these are opportunity costs to the Indian govenrment in lost revenue, but the Indian govenrment is not spending resources prosecuting them either. Alternatively, a false positive means that the Indian govenrment is forced to allocate resources, but there is uncertainty in whether the individual is actually committing fraud or not.

Therefore, I believe that, in this context, it is better for the Indian government to not worry about false negatives as much as false positives because false negatives indicate lost revenue and no cost, but false positives indicate cost incurred by redirecting administrative resources and no increase in tax revenue.

5.

(a)

```
from sklearn.neighbors import KNeighborsClassifier
knn = KNeighborsClassifier(n_neighbors = 5)

df["target"] = np.where(df["Risk"] > .5, 1, 0)
X_train, X_test, y_train, y_test = train_test_split(df.loc[:, "Sector_score":"Audit_Risk"], d


knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print(f"knn confusion matrix:")
print(f"{confusion_matrix(y_test, y_pred)}")
```

```
knn confusion matrix:
[[226   3]
 [ 11 148]]
```

(b)

```
tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
error_rate = (fp + fn) / (tn + fp + fn + tp)
print(f"knn error rate: {round(error_rate, 2)*100} %")
```

```
knn error rate: 4.0 %
```

Our predictions are extremely accurate, predicting correctly with 96% accuracy.

(c)

```
evasion = tp / (fp + tp)
print(f"{round(evasion, 2) * 100} % of firms that were predicted to evade their taxes actuall
```

```
98.0 % of firms that were predicted to evade their taxes actually evaded their taxes
```

6.

(a)

```
# scaling the dataset, by subtracting away the mean and dividing by standard deviation

df_copy = df.copy()
for column in df_copy.columns:
    # using z-score scaling
    df_copy[column] = (df_copy[column] - np.mean(df_copy[column])) / np.std(df_copy[column])


X_train, X_test, y_train, y_test = train_test_split(df_copy.loc[:, "Sector_score":"Audit_Risk

knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)
print("confusion matrix:")
print(confusion_matrix(y_pred, y_test))
```

```
confusion matrix:
[[230  21]
 [  5 132]]
```

(b)

```
tn, fp, fn, tp = confusion_matrix(y_pred, y_test).ravel()
print(f"knn scaled error rate:")
print(f"{round(error_rate, 2) * 100} %")
```

```
knn scaled error rate:
4.0 %
```

Our predictions have a 96% accuracy rate.

(c)

```
evasion = tp / (fp + tp)
print(f"{round(evasion, 2) * 100} % of firms that were predicted to evade their taxes actuall
```

```
86.0 % of firms that were predicted to evade their taxes actually evaded their taxes
```

7.

The scaled data gives us much better predictions in that it reduces the number of false negatives, as well as slightly reducing the number of false positives. Therefore we are interested in using the scaled KNN model instead of the non-scaled KNN model.

I believe that the reason for the better predictions in the scaled KNN model is that KNN is based on the distance of each data point relative to the group. Therefore, in the non-scaled dataset, some of the features are on completely different scales relative to other features. Therefore, the distance between one feature and the next will be huge where as in the scaled model, all features are at the same scale of standard deviations away from the mean. Therefore we are able to get "more information" across all of our features instead of over relying on specific features that will bias our predictions.

8.

```
from sklearn.model_selection import cross_validate

features = df_copy.loc[:, "Sector_score":"Audit_Risk"]
target = df.loc[:, "target"]

scores = cross_validate(knn, features, target, cv = 5)
print(f"cv scores/accuracy array (test):{scores['test_score']}")
print(f"mean cv score/accuracy (test): {round(np.mean(scores['test_score']), 2) * 100}%")
print()
print(f"error rates (test): {1- scores['test_score']}")
print(f"mean error rate (test): {100-round(np.mean(scores['test_score']), 2) * 100}%")
```

```
cv scores/accuracy array (test):[0.97419355 0.95483871 0.94193548 0.89032258 0.82580645]
mean cv score/accuracy (test): 92.0%

error rates (test): [0.02580645 0.04516129 0.05806452 0.10967742 0.17419355]
mean error rate (test): 8.0%
```

We found that the first k (k = 1) yields the lowest error rate, an error rate of ~2.5%

9.

The "Risk" column is produced by determinig whether the firm evaded taxes as a result of an audit. Therefore, the dataset that we have is limited to firms that were deemed to be audit worthy in the first place, and therefore contains firms that were audit worthy + did or did not commit tax evasion. What we dont have in this dataset is the individuals who were not audit worthy, but DID commit tax evasion. Therefore, the current model that we have will get better at predicting thsoe firms which we already thought were commiting tax evasion. We will not get better at finding cases which we did not expect, or would look good on these metrics but are in fact evading taxes.