Statistical Analysis and Prediction for Bank's Campaign of Opening a Term Deposit

Xiaoting (Theresa) Liu

Table of Contents

Introduction

Big data is now rapidly developing and permeating to all industry. With the analysis technology, the bank can provide better service to their customers and serve beyond their competitors. This can not only increase the stickiness of the banks' existing customers, but also become an important tool to attract new markets. The strategy of traditional commercial banks is to determine their goals and business strategies based on market research and consumer demands by predicting by future economic conditions, combining existing resources and relying on decision-makers experiences. In the era of big data, the use of big data analytics allows banks to perform a more scientific and reasonable assessment of consumer demands and position their services to the right target markets and areas. Its usage provides a strong decision-making support to the internal management, business development and customer marketing. In this paper, I am going to analyze a bank data to discover insights from various factors contributed to term deposits acceptance. Distinctive models will be applied to discover the relationship between the factors. Lastly, I will provide suggestion on how to improve the term deposit performance.

Goal of the Paper

- Understand different attributes that affect term deposit acceptance.
- Identify a prediction model to gain customer's interest in opening term deposit.
- Provide suggestions to improve next marketing campaign of term deposit.

Data Description

The data comes from a Portuguese banking institution and retrieved from UCI Machine Learning Repository. There are total 17 inputs and 11,162 records provided in the dataset. The main purpose of this dataset is to discover the classification to predict if the customer will accept/reject a term deposit, in which the term deposit acceptance will be considered as variable y in this project. Appendix 1 introduces more information about the attributes from UCI (Moro et al., 2014).

Literature Review

Banking System with Big Data

Traditional banking system has a great potential to achieve a higher level of operation by integrating technology, which is called modern banking system. The allure of modern banking lies on its availability and convenience that allow customers to control their accounts remotely and independently. Modern banking system has a unique advantage on its data volume and variety that allow us to discover the trends and patterns for future product development (Mungai & Bayat, 2018). In a survey conducted in July 2018, most of the participants claims that they felt the banks could do better at educating them about products and services (Dicamillo, 2019). Banking system now are desiring to grow with the help of data analytics. Today, banks' innovation practices are permeating their organizations, enabling all to collaborate with clients in exploring new opportunities (Atak, 2018). Corsman (2018) also mentioned that on the investment banking side, Morgan Stanley does a lot of tech M&A, and that the bank's technology organization helps the investment bankers understand tech clients and their products, making sure they know what is ahead.

Big Data with Marketing Campaign

Mungai and Bayat (2018) states that "the use of big data by financial service provider has long been a key success factor." They also indicated all major banks in South Africa have embarked on big data projects then thus results 83% increase in clients over past five years (Mungai & Bayat, 2018). A case study named "Wells Fargo Bank Uses Big Data to Increase Loan Recoveries" posted on Management Consulting Case Interviews is taking Wells Fargo as an example to explain they use "big data" analysis to identify abnormal customer behaviors as risk warning signals. The data used for the analysis is the massive transaction data of the bank itself, namely the individual payment data or enterprises transactional data. In its post-loan management, "big data" analysis is helping the bank to optimize collection management. Through a quantitative analysis, the analysis found that nearly 30% of failure collection originated from the failure to contact the borrower. Thus, ensuring borrowers contact information became an important factor to improve the business success. The problem for many banks is to consolidate and access data scattered across various sectors. In terms of "coordination", the financial institutions often have to face the problem of poor communication between business and technology, which makes it difficult to turn data into productivity. We believed that "big data" is more significant in its ability to drive embedded changes and provide insights to better improve banking system performance.

The Importance of Customer Analysis to Banking Industry

Banking system have become more efficient than ever, particularly when using all available data to drive new revenue and growth. Shirazi and Mohammadi (2019) use big data analytics to identify the key factors and create a model for customer churn prediction in retiree segment. Beside maintaining a good relationship with customer, understanding customers' preference is also a significant factor. A research paper provides a report indicates that a new service launch in a company can reach failure rate of 80% in the financial service industry (Meigounpoory, M. R., & Saffari, E., 2014). This study also discloses the report shows 30% of new service development projects of service firms did not meet objectives, caused by the lack of an efficient development process (Alam and Perry, 2002; De Brentani, 1991), and the shortage of customer orientation and input (Martin and Horne, 1995). IBM also claims that personalized service, real-time solutions and the ability to do business is what customers are looking for now. To be successful in banking industry, it is important the bank can offer various products to meet the demands. Customer analysis is the key to maximize the bank development by analyzing customer data. Ernest & Young provides some key findings that show customers evaluate many factors in creating relationship with the bank (2018).

Exploratory and Descriptive Analysis

In this section, I am going provide descriptive analysis to better understand the dataset. Exploratory data analysis enables me to detect outliners and missing values and explore the relationship between characteristic variables, and the relationship between characteristic variables and the target variables. This dataset contains both numerical and categorical data. I will examine for both types separately to find out the important attributes to term deposit subscription.

Numerical Variables. There are 7 numeric attributes shown in below scatter plot, where green dots indicate term deposit acceptance and red dots indicate term deposit rejection. Here are my findings from the below scatter plot (Appendix 2):

- There is no significant relationship between term deposit subscription with age group, balance in the bank, last contract date, number of contacts performed during this campaign and for this client, and the number of contacts performed before this campaign and for this client.

- For the attributes "duration" and "pdays", I found out that there is difference between the term deposit subscription. The heatmap (Appendix 3) indicates "duration" has a positive relationship with term deposit subscription (Recall that "duration" indicates last contact duration, and "pdays" indicates the number of days that passed by after the client was last contacted from a previous campaign).
- From the stacked bar plot (Appendix 4), I can see that clients (77%) who are contacted above average duration are more likely to subscribe a term deposit.

Categorical Variables. There are some categorical variables such as job type, marital status and education level, ect. I did an exploratory analysis to better understand how these categorical variables related to term deposit subscription. Here are my findings (see Appendix 5):

- Job type "management", "retired", "student", and "technician" are more likely to open a term deposit account, and most of the clients are working under "management" type of jobs.
- Clients who are single and have tertiary education are more likely to subscript a term deposit.
- Clients who have housing loan and personal loan tend to reject term deposit subscription; while clients who do not have housing loan are more likely to subscript term deposit.

Predictive Analysis

Predictive analysis utilizes machine learning algorithms to detect what will happen to future term deposit subscription. Based on the large amount of subscription history, the program is able to run different models to help recognizing the trend, customer behavior and insights. This dataset is a hybrid dataset which contains both numerical data and categorical data. The dataset has to be processed before building the model. The “yes and no questions” attributes are encoded to binary class 0 or 1 (No or Yes); while different values starting from 0 are assigned to those attributes that have different class. The predictive analysis analyzed 80% of the dataset. After identifying the best machine learning model, the rest 20% of the dataset will be used to test the model accuracy. There are total 6 models are selected in the predictive analysis. From the **accuracy score** (Appendix 6) calculated by machine learning, the model with highest accuracy score will be further analyzed. Gradient Boosting Classifier has the highest accuracy score of 0.84; while others have about 0.80 without significant difference.

Gradient Boosting Classifier. This classifier has different performance at different **learning rates**. From the table (see Appendix 7), the learning rate of 0.5 shows the best prediction on the training set and a good prediction of the test set. The **classification report** and **confusion matrix** (Appendix 8) are built based on the best learning rate 0.5. Classification report shows that the model correctly predicts term deposit rejection at 87% of the time and term deposit acceptance at 85% of the time. The confusion matrix shows that the test set has 1,129 actual refused deposit and 1,084 actual accepted deposit. There are 960 refused deposit and 941 accepted deposit are correctly predicted.

There are two important we need to understand before the next step: sensitivity and specificity. In terms of the y variable in this dataset, the proportion of customers that are identified correctly to subscribe term deposit upon the total customers who actually subscribe term deposit is called sensitivity or recall. Similarly, the proportion of customers that are identified correctly to not subscribe term deposit upon the total customers who actually do not subscribe term deposit is called specificity or precision. The **sensitivity and specificity tradeoff** plot (Appendix 9) help to identify the threshold to get more term deposit acceptance value predicted correctly. **Receiver**

operating characteristic curve (ROC) (Appendix 10) is a comprehensive indicator of the sensitivity and specificity of continuous variables. The **area under the curve (AUC)** illustrates how well the model performances. The model has AUC score of 0.91. Consider all statistic I get, this model has pretty good performance to predict term deposit acceptance separated from the term deposit rejection.

XGBoost Algorithm. XGBoost is an ensemble machine learning algorithm based on decision tree, and it is used as part of gradient boosting framework (Morde, 2019). Morde (2019) also indicates that the reason why XGBoost has better performance is because it improves "improves upon the base GBM framework through systems optimization and algorithmic enhancements". There are various benefits the classifier can provide such as parallelized tree building and regularization for avoiding overfitting (see Appendix 11). XGBoost accuracy score (0.86) is higher than Gradient Boosting Classifier (0.84) in training set. Looking at the ROC curve and AUC score (Appendix 12), Gradient Boosting Classifier is slightly higher than XGBoost; however, XGBoost "has the best combination of prediction performance and processing time compared to other algorithms" (Morde, 2019).

Feature Selection. XGBoost classifier is selected for this dataset. The top 5 important features are 'contact', 'poutcome', 'housing', 'duration' and 'month' (see Appendix 13). Based on the features ranking, the bank can make changes and know where to focus to identify its target market.

Prescriptive Analysis

There are many factors that will affect customer's term deposits subscription. Through the findings mentioned in above analysis, there are some important attributes are ranked based on the best machine learning algorithm.

1. Contact communication type along with contact duration is important features to term deposit subscription. Clients who are being contacted through cellular and contacted duration above average have higher chance to subscript a term deposit. The bank should encourage the staff to call the client though the cell phone and try their best to stay chat with the clients as long as possible, and make sure to call client's cell phone first. The staff who contacts the clients can introduce some new campaigns that may interest the clients or provide benefits to their bank portfolio.
2. Outcome of the previous marketing campaign is also an important feature to term deposit subscription. Clients who have successful outcome of the previous marketing campaign are likely to subscribe a term deposit. The bank should identify and create a "VIP" list for those clients as a target market for term deposit subscription.
3. Clients who have no housing loan are more likely to subscript a term deposit. While consider housing loan, the bank can take the client bank balance and personal loan under consideration. Clients who have low balance and loans might not be an appropriate target market to promo term deposit subscription. The next campaign should focus more on individuals who have higher balance and no loan.
4. May seems like to have significant more rejection than other months. February to April, September, October and December are the months that have higher chance to get clients to subscribe a term deposit. The bank can evaluate the reason why May doesn't attract clients to

subscript a term deposit. Are there any other campaigns that offer better interest rate than term deposit in May? If that is the situation, the bank should avoid launching this campaign in May and promote it in other months such as March or September.

Conclusion

The value of big data is not only reflected in the direct impact on the financial indicators, but also in the promotion and reconstruction of business model changes. The bank should give emphasis on the volume, variety, velocity and veracity of big data; while it also needs to carry out in-depth data integration, form its own data assets management ability, and thus generate insight from the data. This project provides deep analysis to identify the key attributes to term deposit subscription. Various analysis and machine learning algorithm help to achieve the goal. By selecting the suitable machine learning model and feed the bank data, the bank is able to make changes and suggestions to improve future marketing campaign on term deposit subscription.

Reference

Alam, I. and Perry, C. (2002). A Customer-Oriented New Service Development Process. Journal of Services Marketing. 16 (6): 515-534

Atak, G. (2018). The secret to successful innovation? Collaboration. Global Finance, 32(10), 49. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=133222499&site=ehost-live&scope=site

Crosman, P. (2018). Morgan Stanley kicks new tech strategy into gear. American Banker, 183(84), 1. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=129367857&site=ehost-live&scope=site

De Brentani, U. (1995). New Industrial Service Development: Scenarios for Success and Failure. Journal of Business Research. 32: 93-103

Dicamillo, N. (2019). Bank vendors hungry for small-business analytics. American Banker, 184(130), 1. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=137366673&site=ehost-live&scope=site

Ernest & Young (2019). Understanding Customer behavior in retail banking. Retrieved from https://www.ey.com/Publication/vwLUAssets/Understanding_customer_behavior_in_retail_banking_-_February_2010/$FILE/EY_Understanding_customer_behavior_in_retail_banking_-_February_2010.pdf

IBM CUSTOMER ANALYTICS FOR BANKING (n.d). Retrieved from https://www.ibmbigdatahub.com/interactive/ibm-customer-analytics-banking

Martin, C. R. Jr. and Horne, D. A. (1995). Level of Success Inputs for Service Innovations in the Same Firm. International Journal of Service Industry Management. 6 (4): 40-56

Meigounpoory, M. R., & Saffari, E. (2014). Effective Factors of Customer Involvement in the Launching of New Services in Banking Systems. International Journal of Management, Accounting & Economics, 1(4), 284–294. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=99263180&site=ehost-live&scope=site

Morde, V. (2019). XGBoost Algorithm: Long May She Reign! Retrieved from https://towardsdatascience.com/https-medium-com-vishalmorde-xgboost-algorithm-long-she-may-rein-edd9f99be63d

Mungai, K., & Bayat, A. (2018). The Impact of Big Data on the South African Banking Industry. Proceedings of the International Conference on Intellectual Capital, Knowledge

Management & Organizational Learning, 225–236. Retrieved from http://search.ebscohost.com/login.aspx?direct=true&AuthType=sso&db=bth&AN=136627116&site=ehost-live&scope=site

S. Moro, P. Cortez and P. Rita. (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014

Shirazi, F., & Mohammadi, M. (2019). A big data analytics model for customer churn prediction in the retiree segment. International Journal of Information Management, 48, 238–253. https://doi.org/10.1016/j.ijinfomgt.2018.10.005

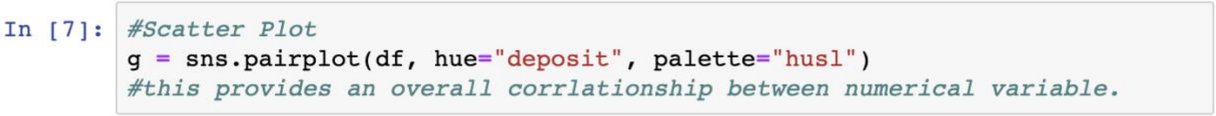
Wells Fargo Bank Uses Big Data to Increase Loan Recoveries. Retrieved from https://www.consultingcase101.com/wells-fargo-bank-uses-big-data-to-increase-loan-recoveries/

Appendix

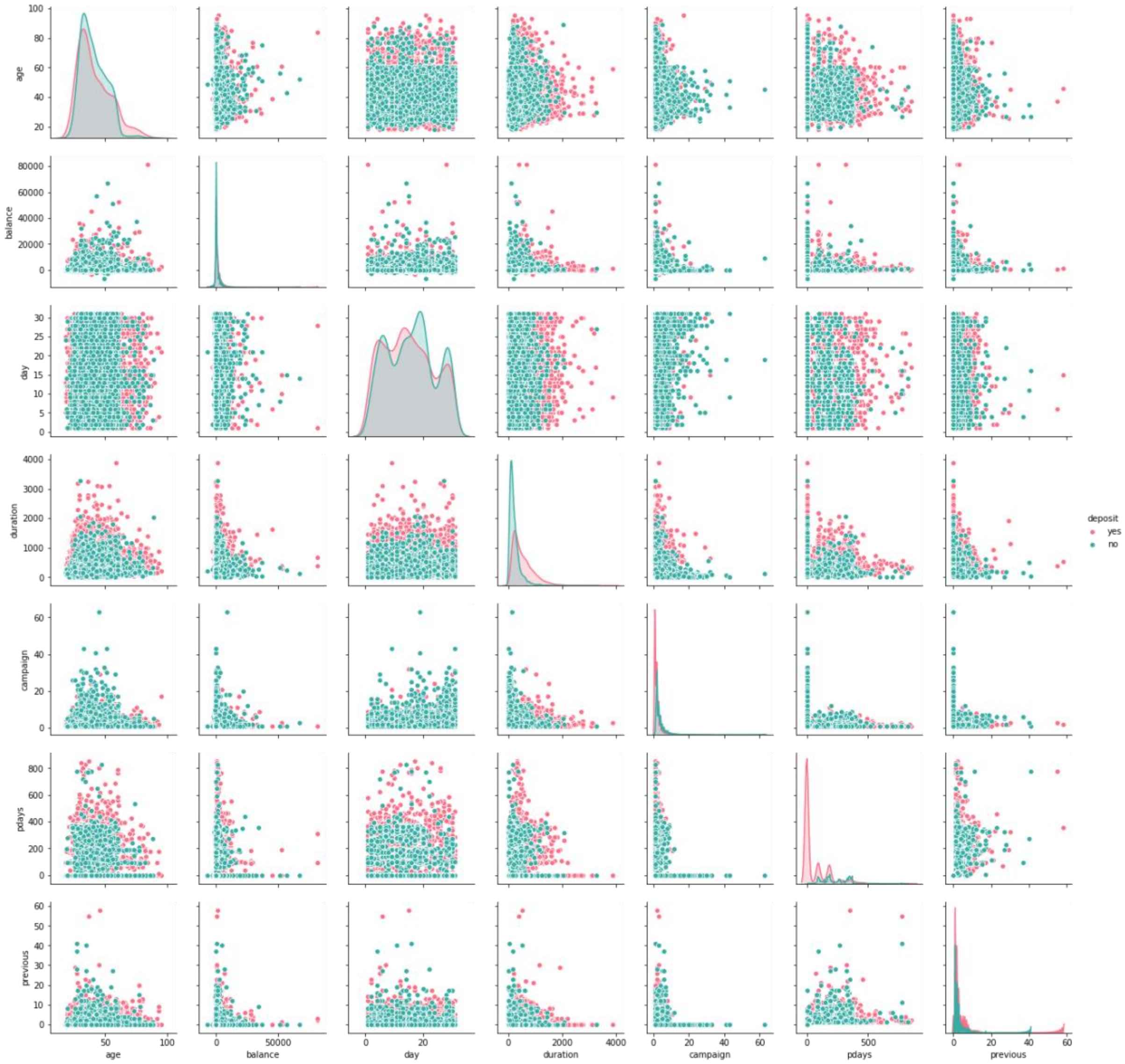
Appendix 1. Attributes Information

Attribute	Description	Range
Age	Age of the clients	Numeric
Job	Type of Job	'admin.','blue-collar','entrepreneur','housemaid','management','retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown'
Marital	Marital Status	'divorced', 'married', 'single', 'unknown'; note: 'divorced' means divorced or widowed)
Education	Categorical	basic.4y','basic.6y','basic.9y', 'high.school', 'illiterate', 'professional.course', 'university.degree', 'unknown')
Default	Has credit in default?	'no', 'yes', 'unknown'
Balance	Money in the bank	Numeric
Housing	Hosing loan	'no', 'yes', 'unknown'
Loan	Personal loan	'no', 'yes', 'unknown'
Contact	Contact communication type	cellular', 'telephone'
Day	Last contact date of the month	Numeric
Month	Last contact month	'jan', 'feb', 'mar', ..., 'nov', 'dec'
Duration	last contact duration, in seconds	numeric
Campaign	number of contacts performed during this campaign and for this client	numeric
Pdays	number of days that passed by after the client was last contacted from a previous campaign	numeric; 999 means clients were not previously contacted)
Previous	number of contacts performed before this campaign and for this client	numeric
POutcome	outcome of the previous marketing campaign	'failure', 'nonexistent', 'success'
Deposit	Has the client subscribed a term deposit?	'yes', 'no'

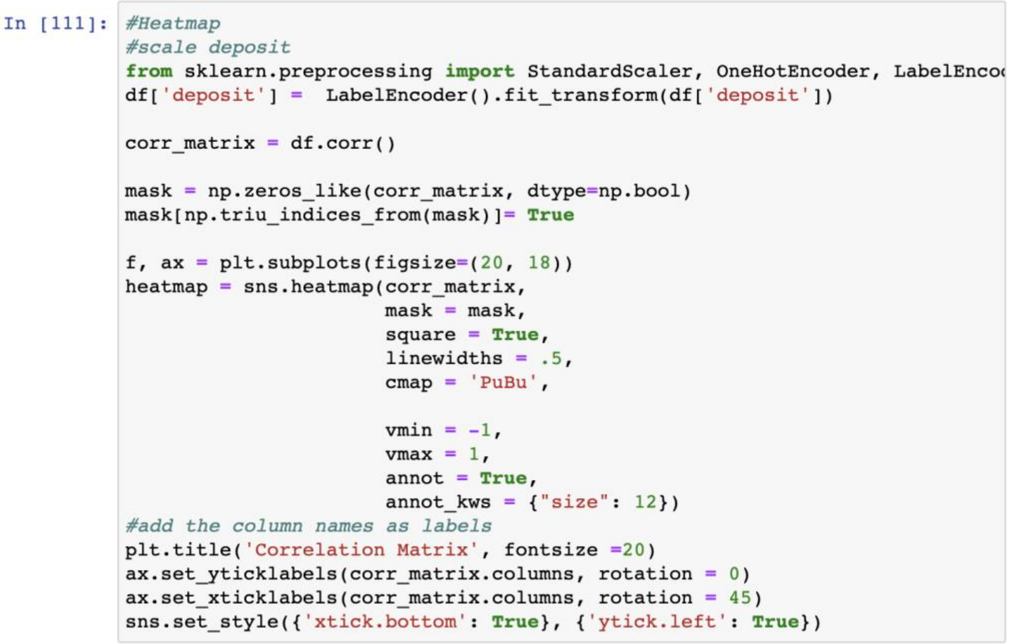
Appendix 2. Scatter Plot



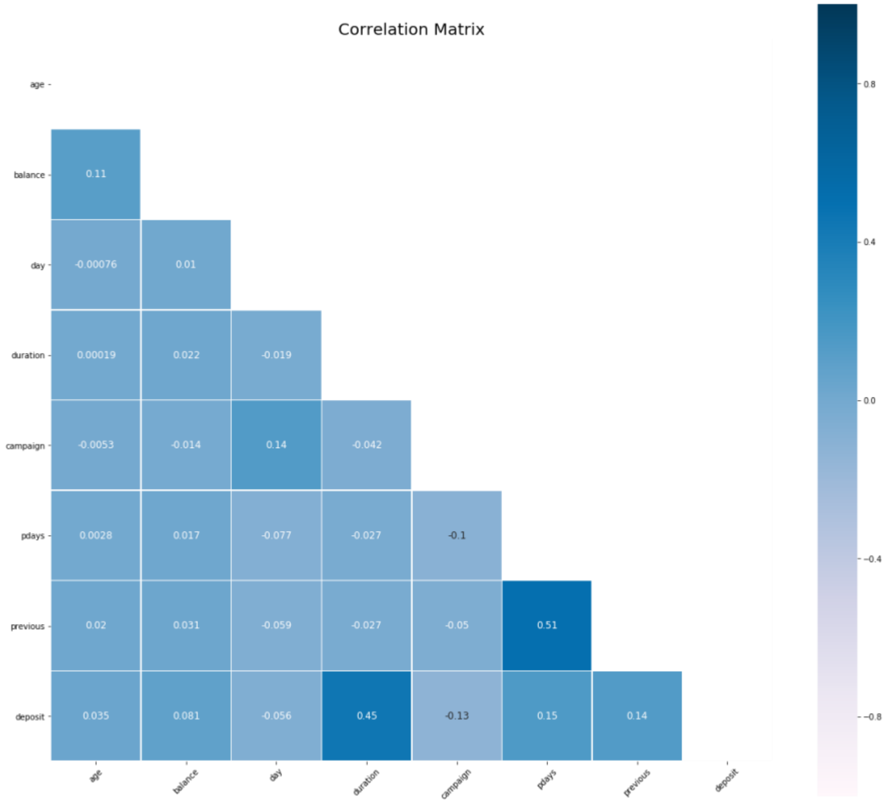
```
In [7]: #Scatter Plot
        g = sns.pairplot(df, hue="deposit", palette="husl")
        #this provides an overall corrlationship between numerical variable.
```



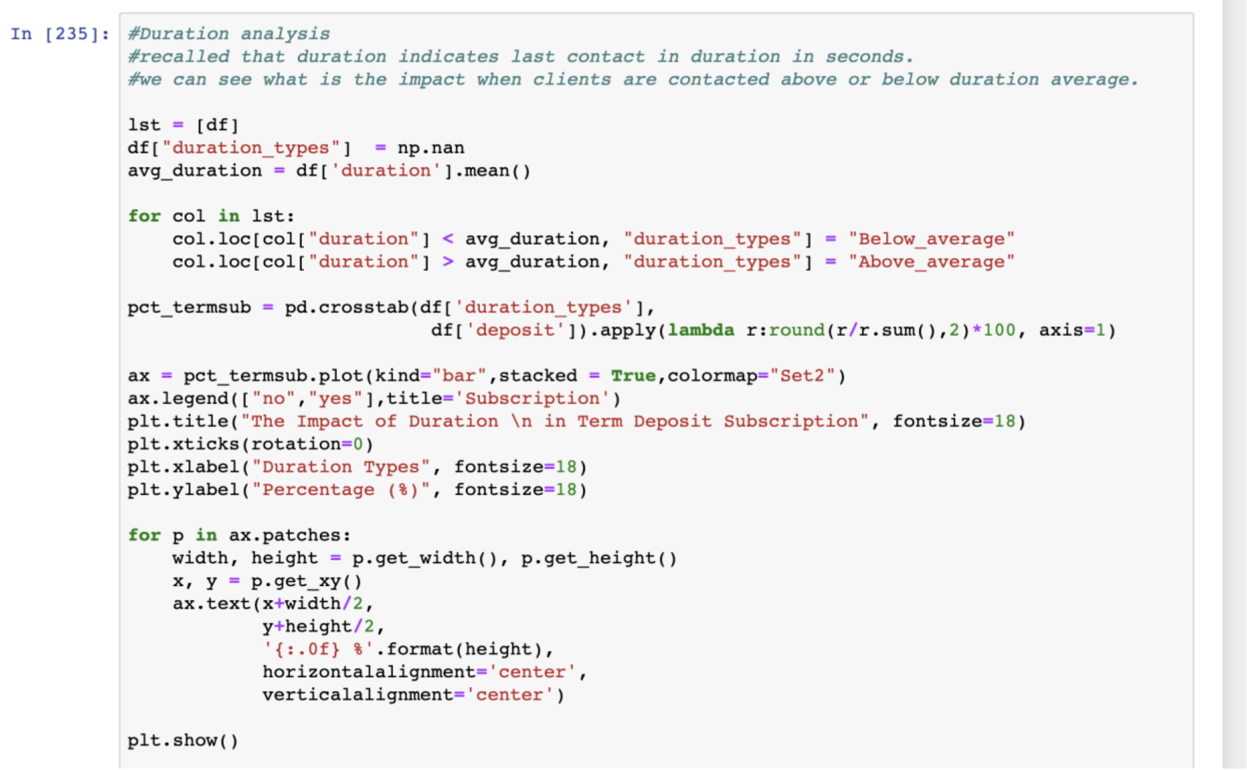
Appendix 3. Correlation Matrix



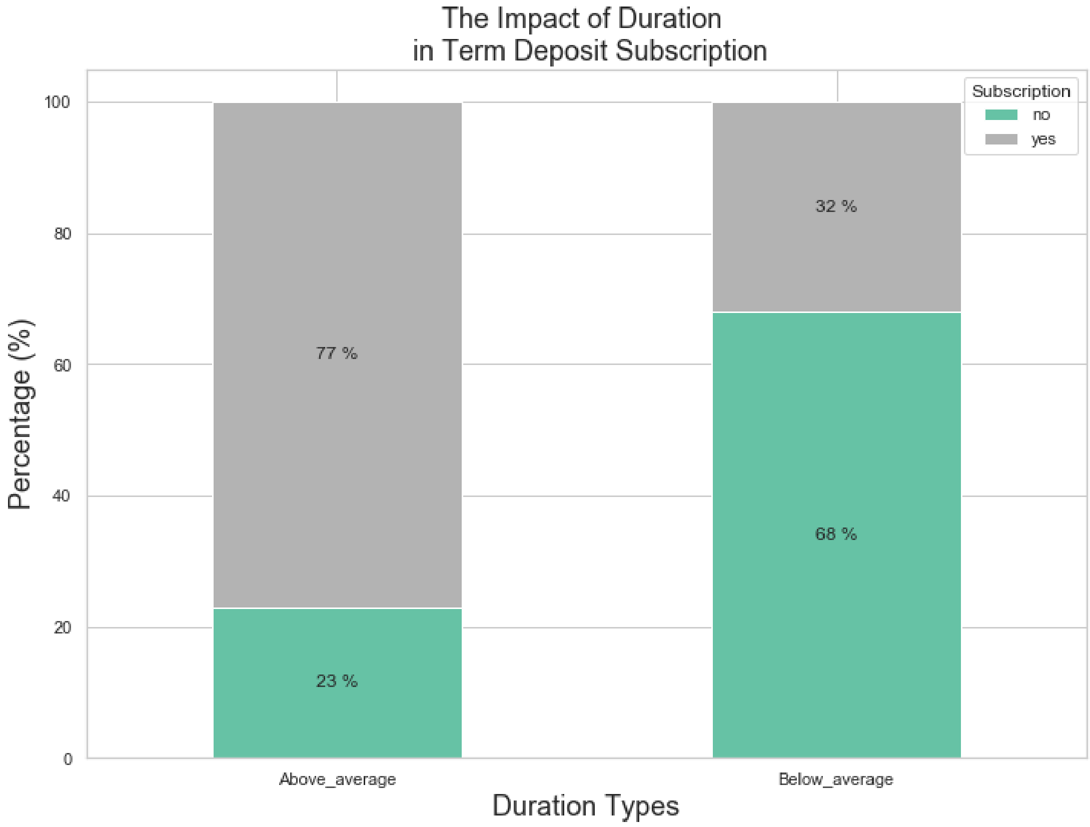
```
In [111]: #Heatmap
          #scale deposit
          from sklearn.preprocessing import StandardScaler, OneHotEncoder, LabelEncod
          df['deposit'] =  LabelEncoder().fit_transform(df['deposit'])

          corr_matrix = df.corr()

          mask = np.zeros_like(corr_matrix, dtype=np.bool)
          mask[np.triu_indices_from(mask)]= True

          f, ax = plt.subplots(figsize=(20, 18))
          heatmap = sns.heatmap(corr_matrix,
                                mask = mask,
                                square = True,
                                linewidths = .5,
                                cmap = 'PuBu',

                                vmin = -1,
                                vmax = 1,
                                annot = True,
                                annot_kws = {"size": 12})
          #add the column names as labels
          plt.title('Correlation Matrix', fontsize =20)
          ax.set_yticklabels(corr_matrix.columns, rotation = 0)
          ax.set_xticklabels(corr_matrix.columns, rotation = 45)
          sns.set_style({'xtick.bottom': True}, {'ytick.left': True})
```



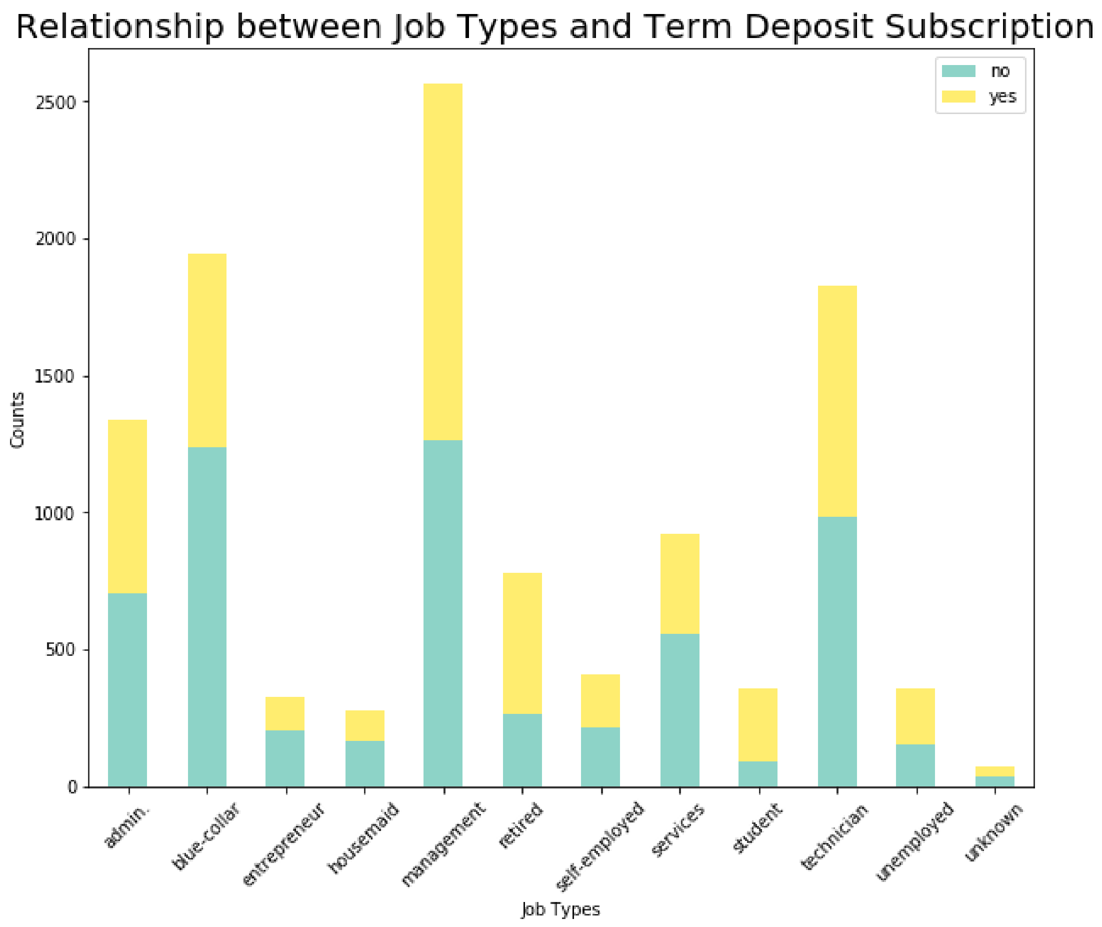
Appendix 4. Duration Analysis

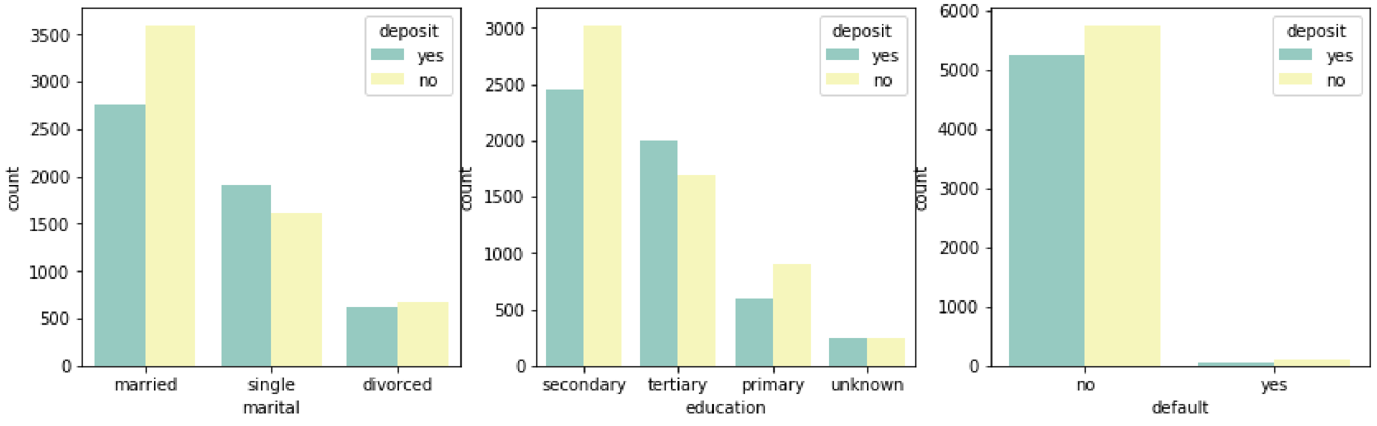


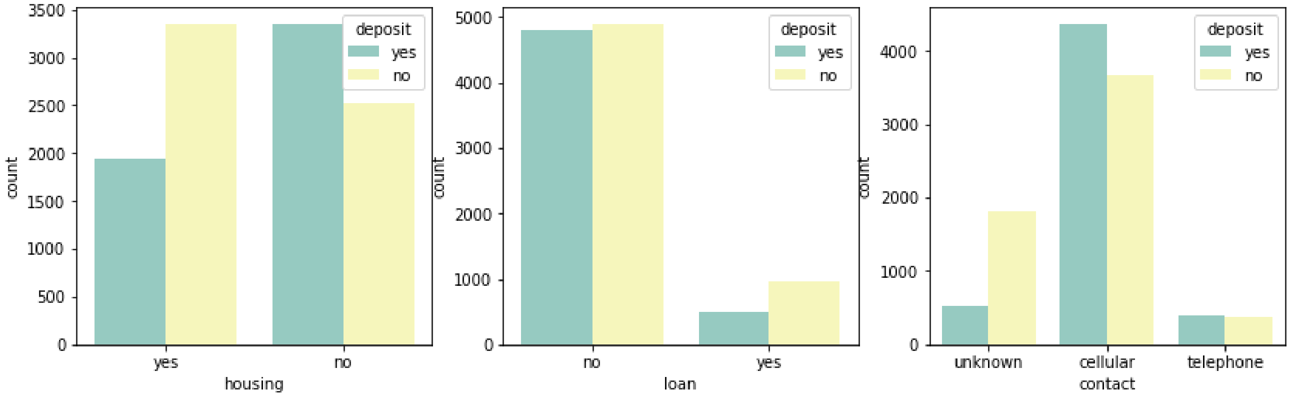
```
In [235]: #Duration analysis
          #recalled that duration indicates last contact in duration in seconds.
          #we can see what is the impact when clients are contacted above or below duration average.

          lst = [df]
          df["duration_types"]  = np.nan
          avg_duration = df['duration'].mean()

          for col in lst:
              col.loc[col["duration"] < avg_duration, "duration_types"] = "Below_average"
              col.loc[col["duration"] > avg_duration, "duration_types"] = "Above_average"

          pct_termsub = pd.crosstab(df['duration_types'],
                                    df['deposit']).apply(lambda r:round(r/r.sum(),2)*100, axis=1)

          ax = pct_termsub.plot(kind="bar",stacked = True,colormap="Set2")
          ax.legend(["no","yes"],title='Subscription')
          plt.title("The Impact of Duration \n in Term Deposit Subscription", fontsize=18)
          plt.xticks(rotation=0)
          plt.xlabel("Duration Types", fontsize=18)
          plt.ylabel("Percentage (%)", fontsize=18)

          for p in ax.patches:
              width, height = p.get_width(), p.get_height()
              x, y = p.get_xy()
              ax.text(x+width/2,
                      y+height/2,
                      '{:.0f} %'.format(height),
                      horizontalalignment='center',
                      verticalalignment='center')

          plt.show()
```

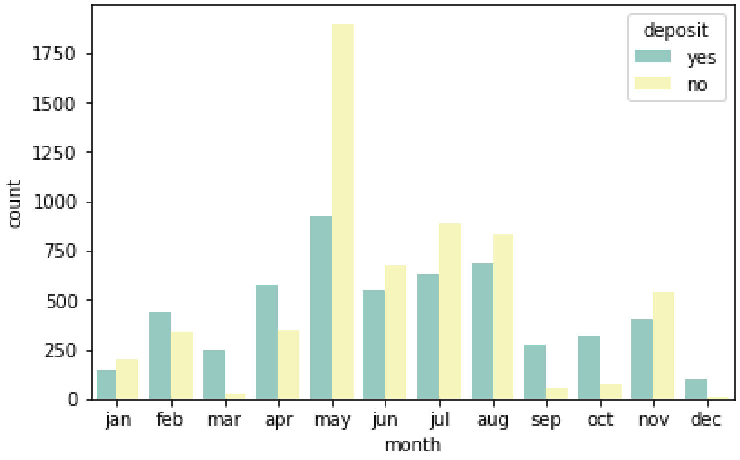


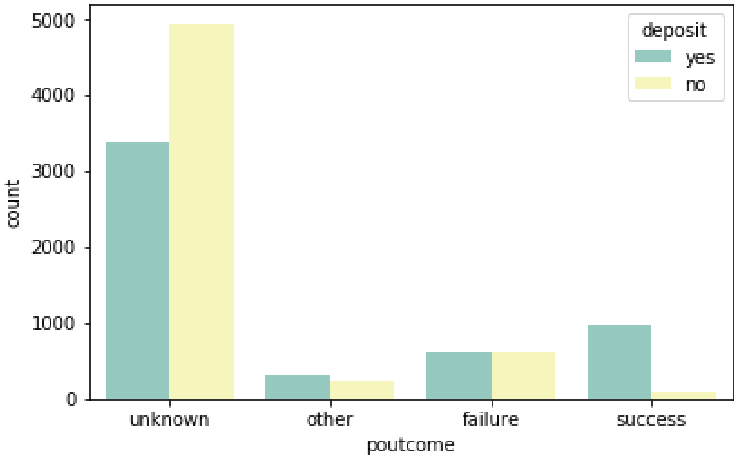
Appendix 5. EDA Analysis



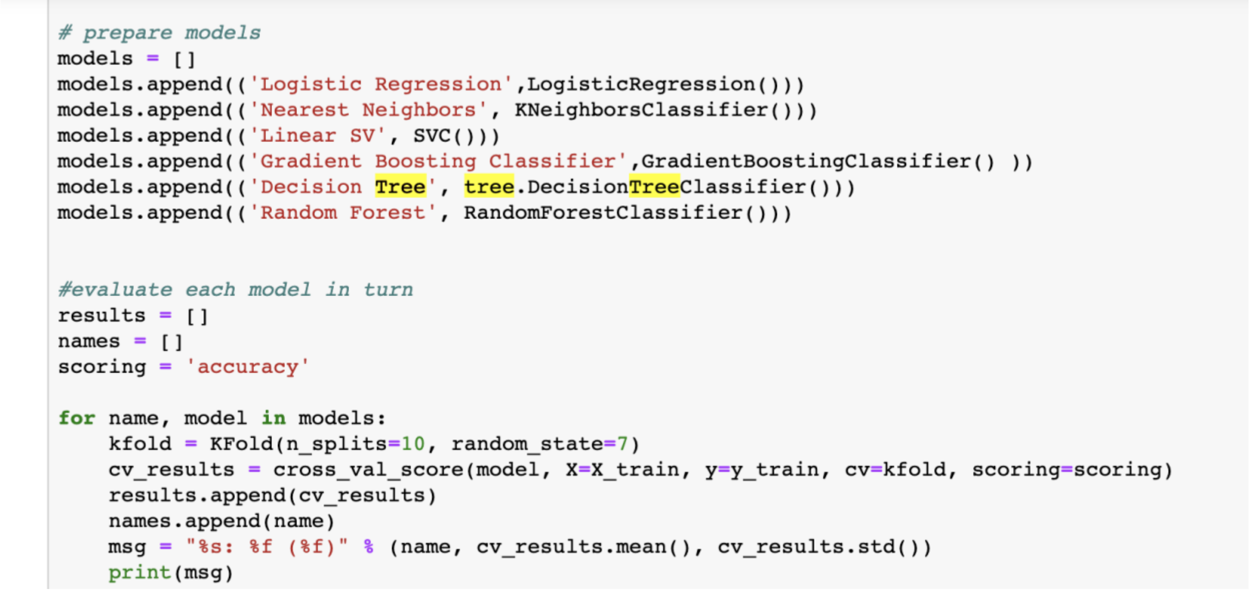



count
3500
3000
2500
2000
1500
1000
500
0
yes
no
housing
deposit
yes
no
count
5000
4000
3000
2000
1000
0
no
yes
loan
deposit
yes
no
count
4000
3000
2000
1000
0
unknown
cellular
telephone
contact
deposit
yes
no

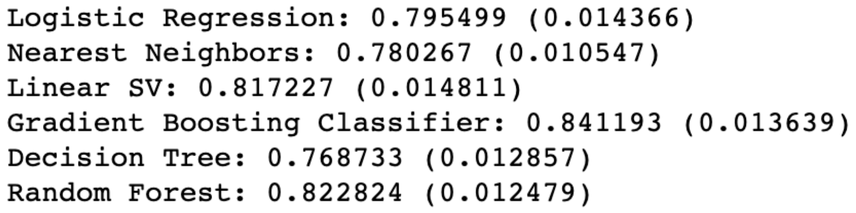

count
1750
1500
1250
1000
750
500
250
0
jan
feb
mar
apr
may
jun
jul
aug
sep
oct
nov
dec
month
deposit
yes
no


count
5000
4000
3000
2000
1000
0
unknown
other
failure
success
poutcome
deposit
yes
no

Appendix 6. Model Selection

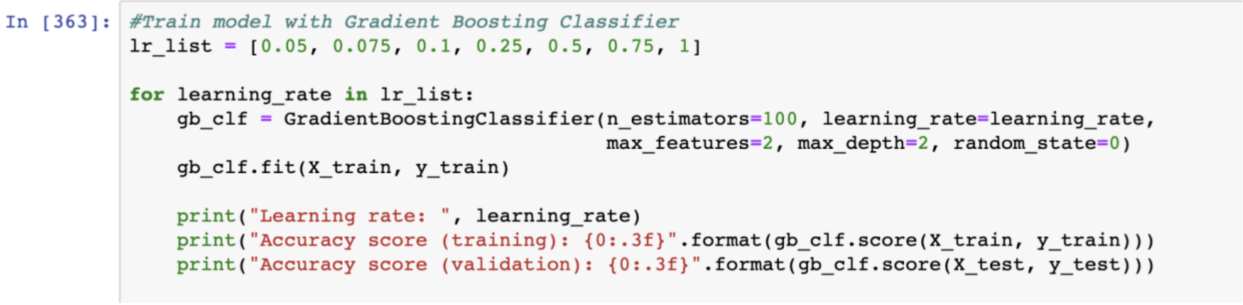


```
# prepare models
models = []
models.append(('Logistic Regression',LogisticRegression()))
models.append(('Nearest Neighbors', KNeighborsClassifier()))
models.append(('Linear SV', SVC()))
models.append(('Gradient Boosting Classifier',GradientBoostingClassifier() ))
models.append(('Decision Tree', tree.DecisionTreeClassifier()))
models.append(('Random Forest', RandomForestClassifier()))


#evaluate each model in turn
results = []
names = []
scoring = 'accuracy'

for name, model in models:
    kfold = KFold(n_splits=10, random_state=7)
    cv_results = cross_val_score(model, X=X_train, y=y_train, cv=kfold, scoring=scoring)
    results.append(cv_results)
    names.append(name)
    msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
    print(msg)
```

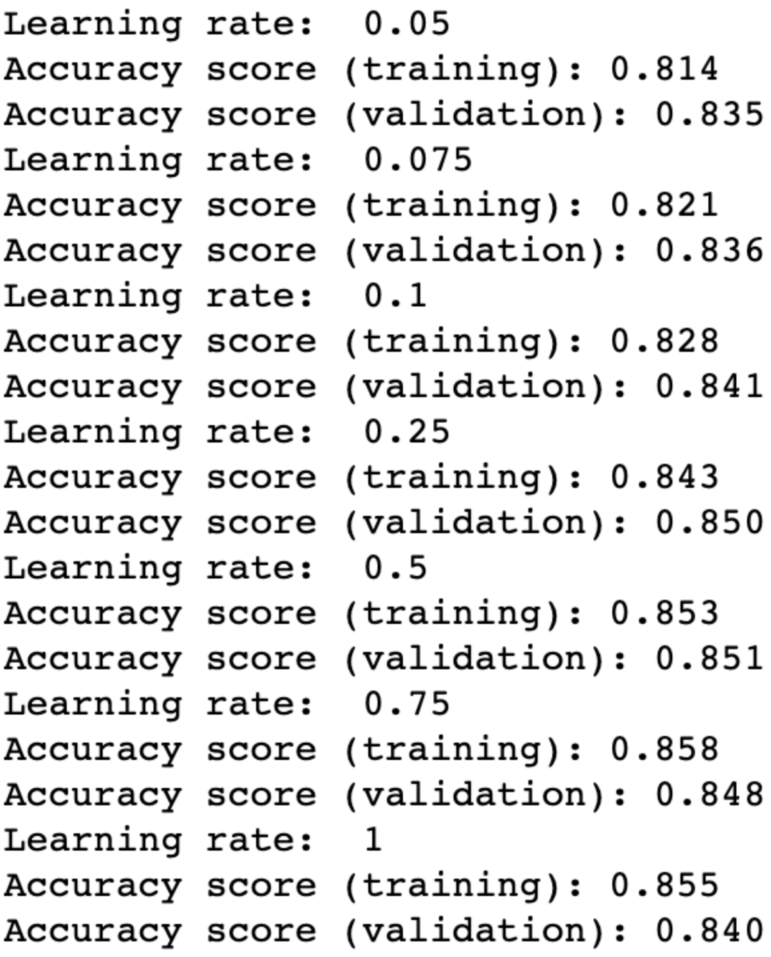


```
Logistic Regression: 0.795499 (0.014366)
Nearest Neighbors: 0.780267 (0.010547)
Linear SV: 0.817227 (0.014811)
Gradient Boosting Classifier: 0.841193 (0.013639)
Decision Tree: 0.768733 (0.012857)
Random Forest: 0.822824 (0.012479)
```

Appendix 7. Calcuate Learning Rate

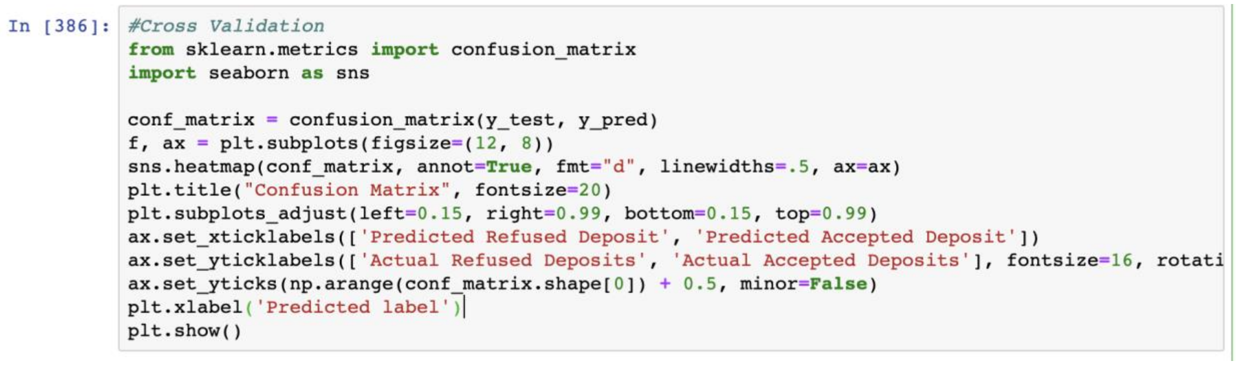


```
In [363]: #Train model with Gradient Boosting Classifier
          lr_list = [0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1]

          for learning_rate in lr_list:
              gb_clf = GradientBoostingClassifier(n_estimators=100, learning_rate=learning_rate,
                                                  max_features=2, max_depth=2, random_state=0)
              gb_clf.fit(X_train, y_train)

              print("Learning rate: ", learning_rate)
              print("Accuracy score (training): {0:.3f}".format(gb_clf.score(X_train, y_train)))
              print("Accuracy score (validation): {0:.3f}".format(gb_clf.score(X_test, y_test)))
```

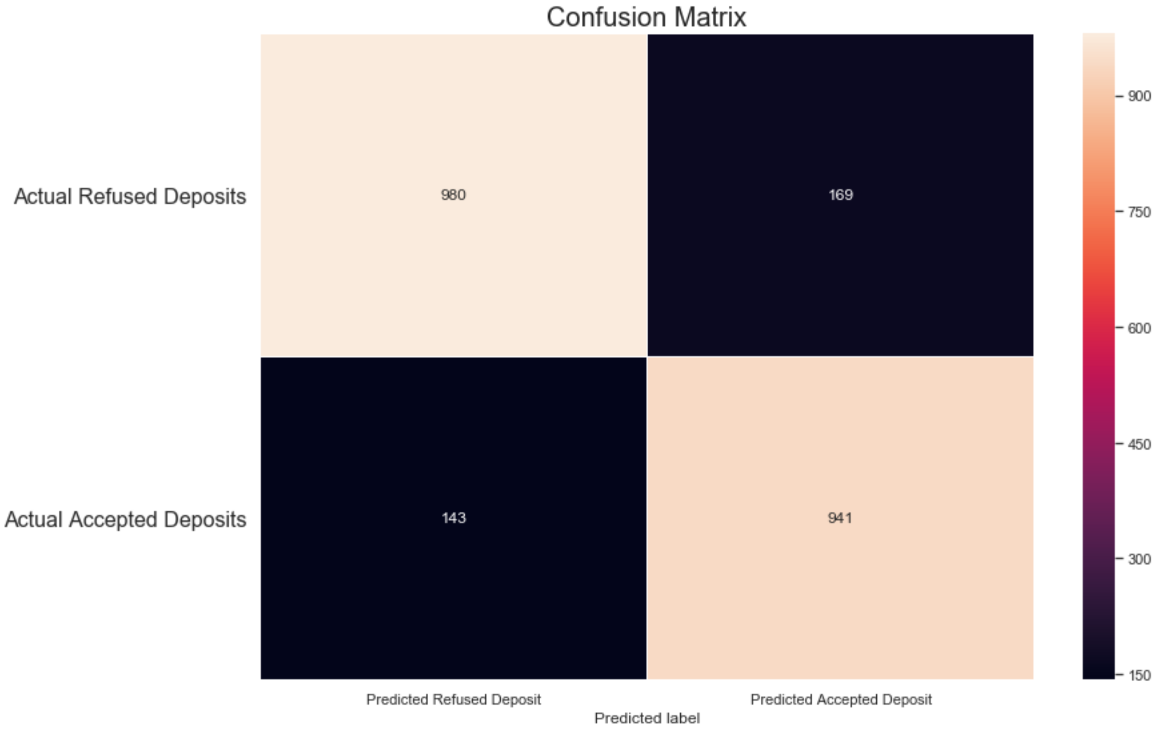


```
Learning rate:  0.05
Accuracy score (training): 0.814
Accuracy score (validation): 0.835
Learning rate:  0.075
Accuracy score (training): 0.821
Accuracy score (validation): 0.836
Learning rate:  0.1
Accuracy score (training): 0.828
Accuracy score (validation): 0.841
Learning rate:  0.25
Accuracy score (training): 0.843
Accuracy score (validation): 0.850
Learning rate:  0.5
Accuracy score (training): 0.853
Accuracy score (validation): 0.851
Learning rate:  0.75
Accuracy score (training): 0.858
Accuracy score (validation): 0.848
Learning rate:  1
Accuracy score (training): 0.855
Accuracy score (validation): 0.840
```

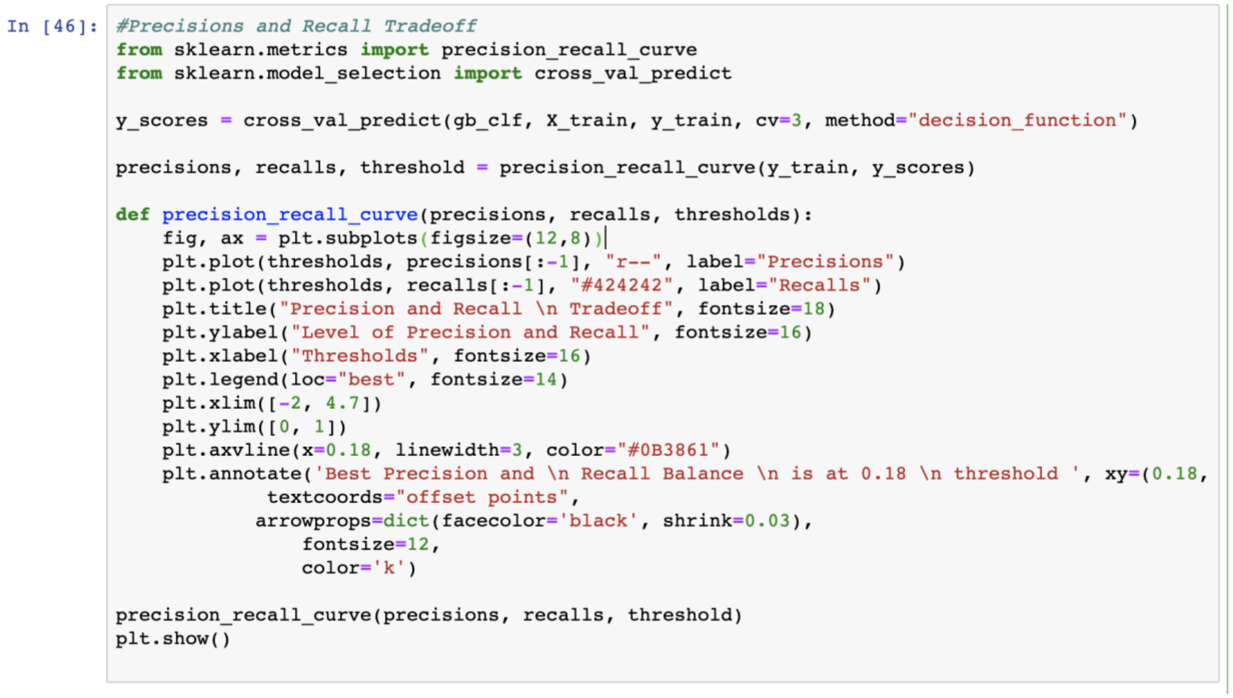
Appendix 8. Cross Validation



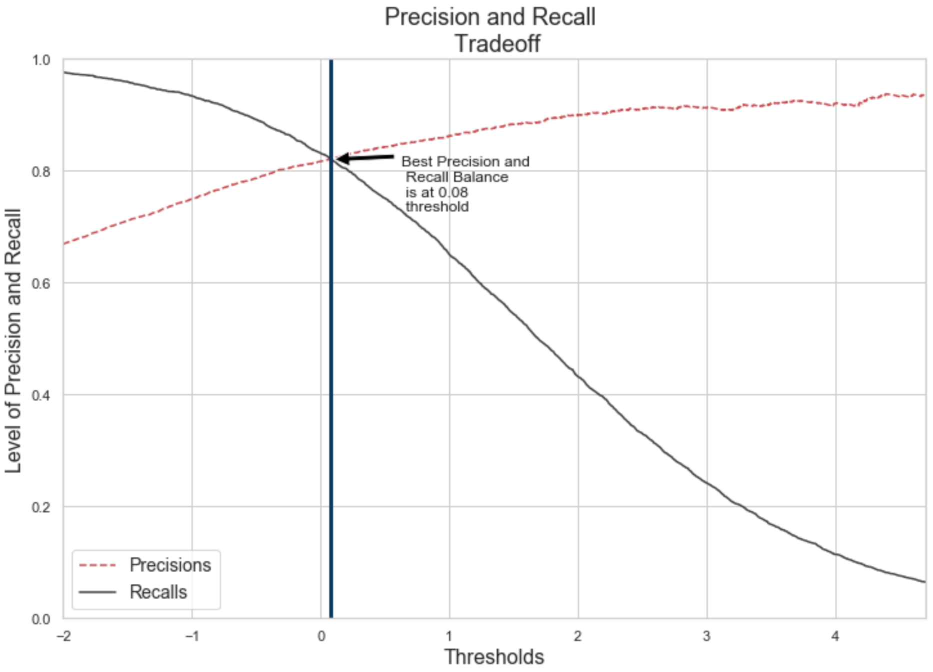
```
In [386]: #Cross Validation
          from sklearn.metrics import confusion_matrix
          import seaborn as sns

          conf_matrix = confusion_matrix(y_test, y_pred)
          f, ax = plt.subplots(figsize=(12, 8))
          sns.heatmap(conf_matrix, annot=True, fmt="d", linewidths=.5, ax=ax)
          plt.title("Confusion Matrix", fontsize=20)
          plt.subplots_adjust(left=0.15, right=0.99, bottom=0.15, top=0.99)
          ax.set_xticklabels(['Predicted Refused Deposit', 'Predicted Accepted Deposit'])
          ax.set_yticklabels(['Actual Refused Deposits', 'Actual Accepted Deposits'], fontsize=16, rotati
          ax.set_yticks(np.arange(conf_matrix.shape[0]) + 0.5, minor=False)
          plt.xlabel('Predicted label')
          plt.show()
```



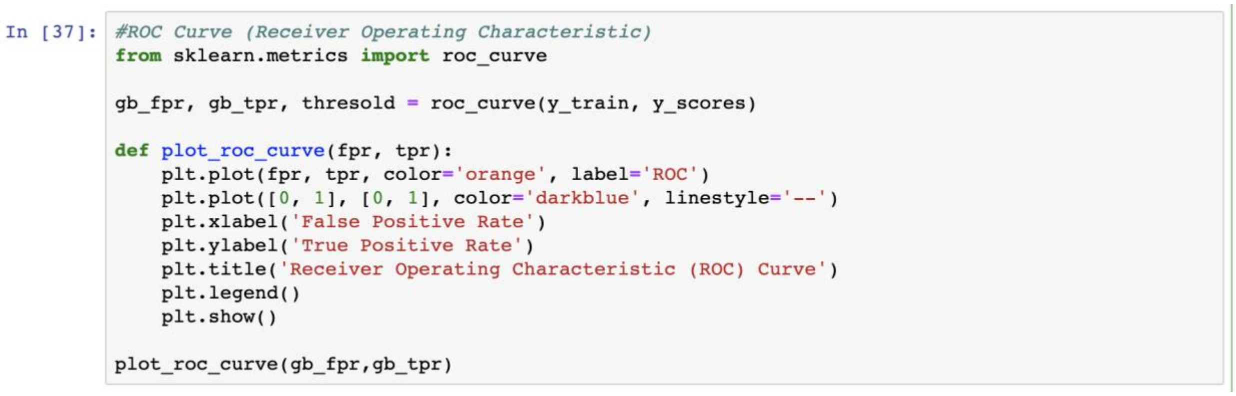
Appendix 9. Precisions and Recall Tradeoff



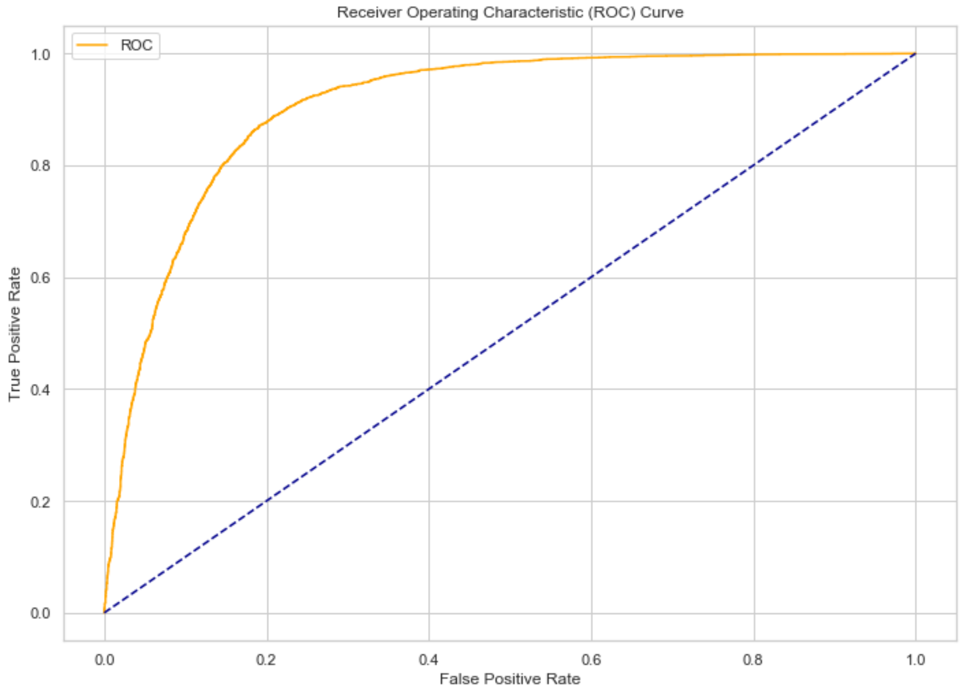
```
In [46]: #Precisions and Recall Tradeoff
         from sklearn.metrics import precision_recall_curve
         from sklearn.model_selection import cross_val_predict

         y_scores = cross_val_predict(gb_clf, X_train, y_train, cv=3, method="decision_function")

         precisions, recalls, threshold = precision_recall_curve(y_train, y_scores)

         def precision_recall_curve(precisions, recalls, thresholds):
             fig, ax = plt.subplots(figsize=(12,8))
             plt.plot(thresholds, precisions[:-1], "r--", label="Precisions")
             plt.plot(thresholds, recalls[:-1], "#424242", label="Recalls")
             plt.title("Precision and Recall \n Tradeoff", fontsize=18)
             plt.ylabel("Level of Precision and Recall", fontsize=16)
             plt.xlabel("Thresholds", fontsize=16)
             plt.legend(loc="best", fontsize=14)
             plt.xlim([-2, 4.7])
             plt.ylim([0, 1])
             plt.axvline(x=0.18, linewidth=3, color="#0B3861")
             plt.annotate('Best Precision and \n Recall Balance \n is at 0.18 \n threshold ', xy=(0.18,
                      textcoords="offset points",
                     arrowprops=dict(facecolor='black', shrink=0.03),
                         fontsize=12,
                         color='k')

         precision_recall_curve(precisions, recalls, threshold)
         plt.show()
```



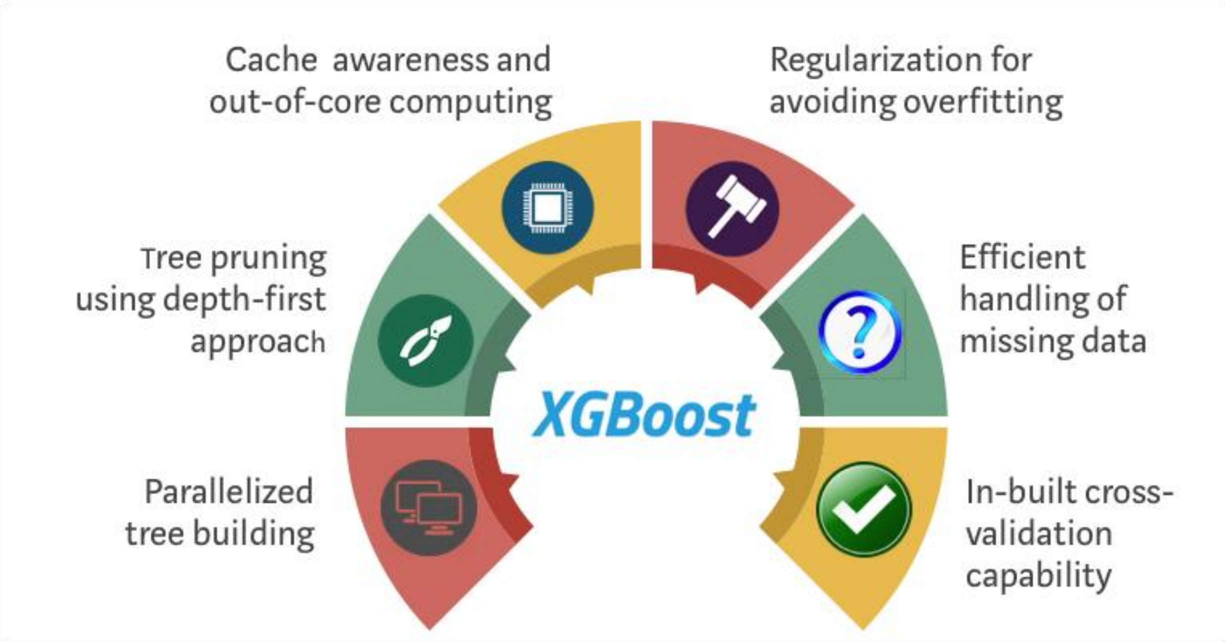
Appendix 10. ROC Curve (Receiver Operating Characteristic)



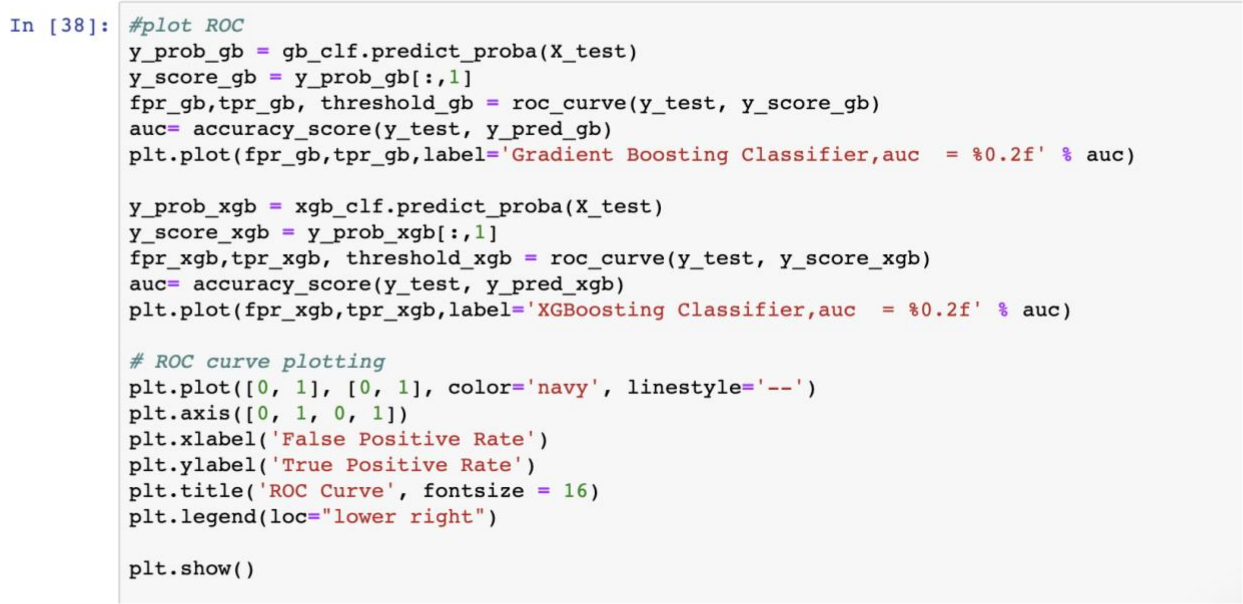
```
In [37]: #ROC Curve (Receiver Operating Characteristic)
         from sklearn.metrics import roc_curve

         gb_fpr, gb_tpr, thresold = roc_curve(y_train, y_scores)

         def plot_roc_curve(fpr, tpr):
             plt.plot(fpr, tpr, color='orange', label='ROC')
             plt.plot([0, 1], [0, 1], color='darkblue', linestyle='--')
             plt.xlabel('False Positive Rate')
             plt.ylabel('True Positive Rate')
             plt.title('Receiver Operating Characteristic (ROC) Curve')
             plt.legend()
             plt.show()

         plot_roc_curve(gb_fpr,gb_tpr)
```



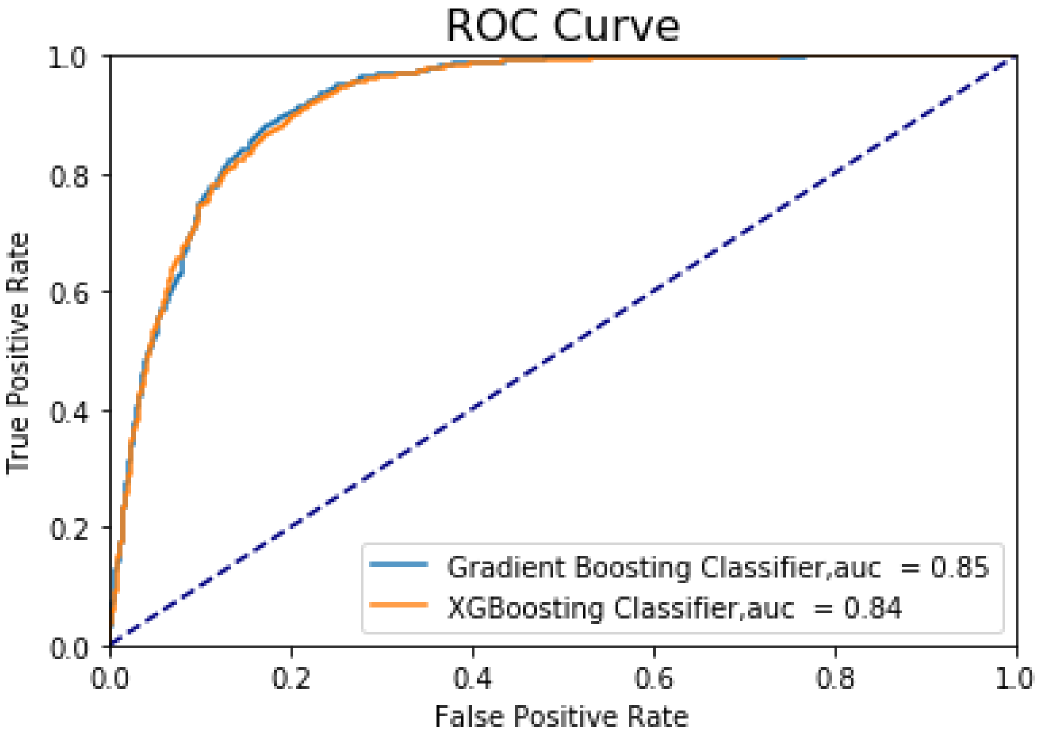
Appendix 11. XGBoost Benefits.


Cache awareness and out-of-core computing
Regularization for avoiding overfitting
Tree pruning using depth-first approach
Efficient handling of missing data
XGBoost
Parallelized tree building
In-built cross-validation capability

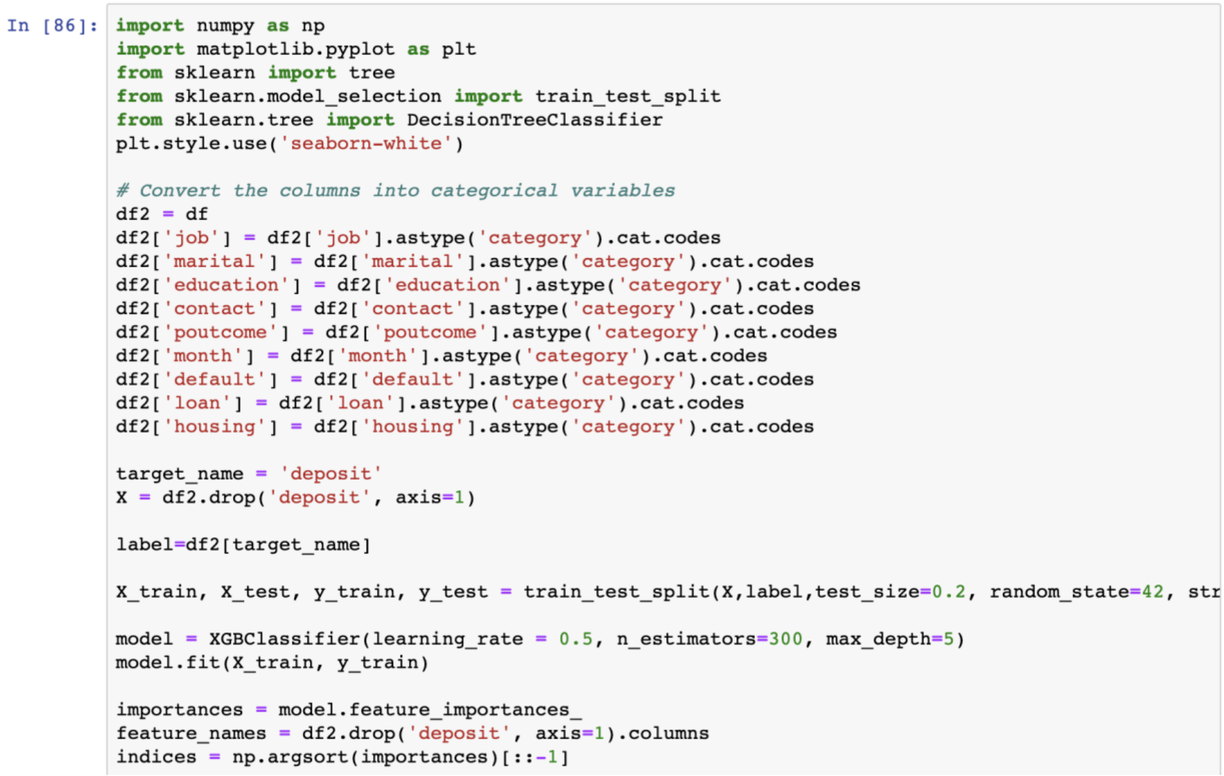
Appendix 12. ROC with XGBoost



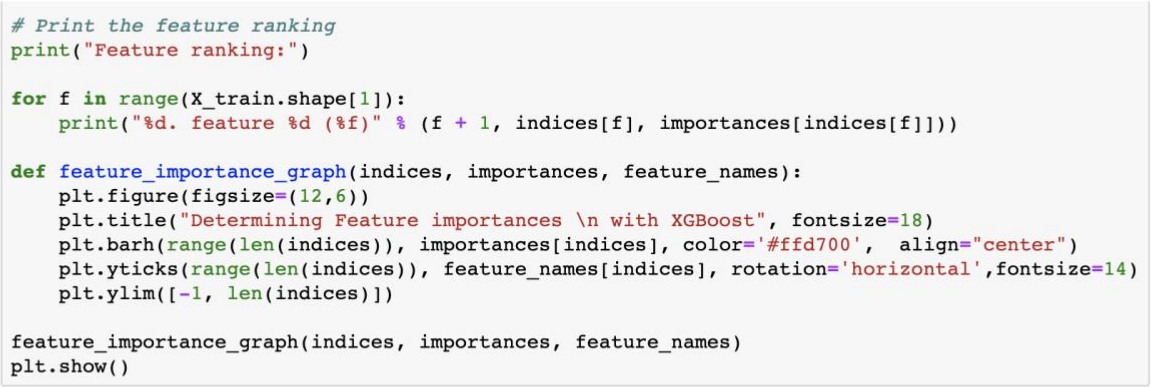
```
In [38]: #plot ROC
         y_prob_gb = gb_clf.predict_proba(X_test)
         y_score_gb = y_prob_gb[:,1]
         fpr_gb,tpr_gb, threshold_gb = roc_curve(y_test, y_score_gb)
         auc= accuracy_score(y_test, y_pred_gb)
         plt.plot(fpr_gb,tpr_gb,label='Gradient Boosting Classifier,auc  = %0.2f' % auc)

         y_prob_xgb = xgb_clf.predict_proba(X_test)
         y_score_xgb = y_prob_xgb[:,1]
         fpr_xgb,tpr_xgb, threshold_xgb = roc_curve(y_test, y_score_xgb)
         auc= accuracy_score(y_test, y_pred_xgb)
         plt.plot(fpr_xgb,tpr_xgb,label='XGBoosting Classifier,auc  = %0.2f' % auc)

         # ROC curve plotting
         plt.plot([0, 1], [0, 1], color='navy', linestyle='--')
         plt.axis([0, 1, 0, 1])
         plt.xlabel('False Positive Rate')
         plt.ylabel('True Positive Rate')
         plt.title('ROC Curve', fontsize = 16)
         plt.legend(loc="lower right")

         plt.show()
```



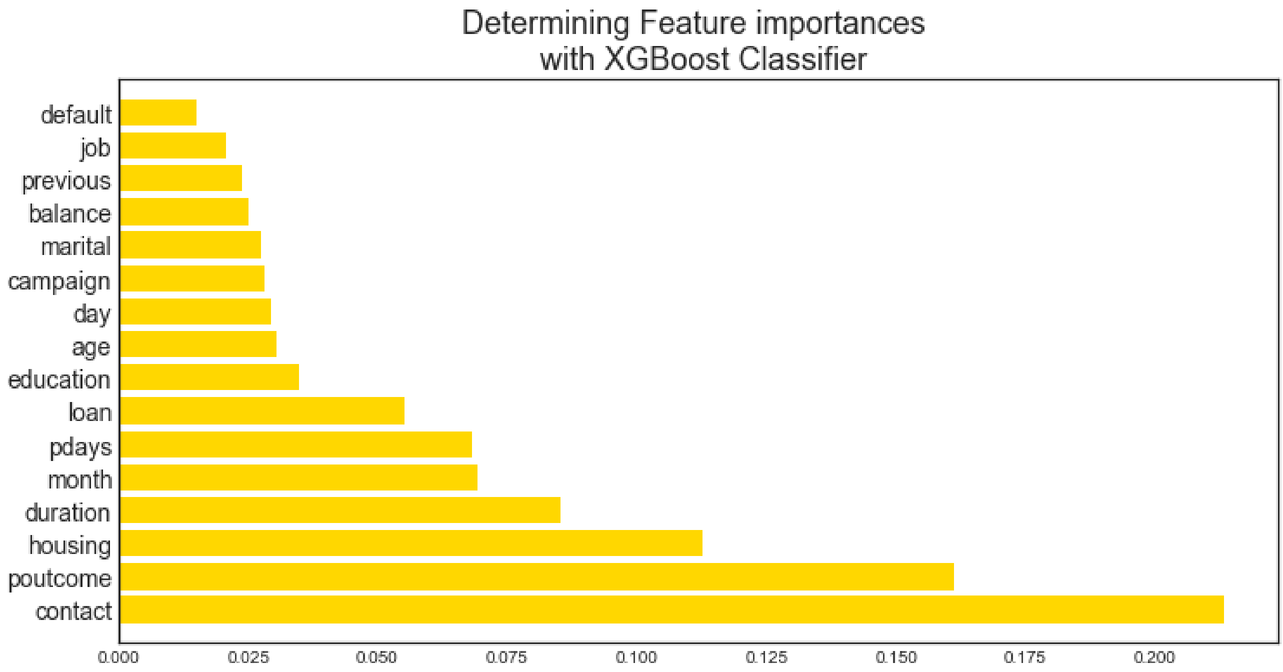
Appendix 13. Feature Selections



```
In [86]: import numpy as np
         import matplotlib.pyplot as plt
         from sklearn import tree
         from sklearn.model_selection import train_test_split
         from sklearn.tree import DecisionTreeClassifier
         plt.style.use('seaborn-white')

         # Convert the columns into categorical variables
         df2 = df
         df2['job'] = df2['job'].astype('category').cat.codes
         df2['marital'] = df2['marital'].astype('category').cat.codes
         df2['education'] = df2['education'].astype('category').cat.codes
         df2['contact'] = df2['contact'].astype('category').cat.codes
         df2['poutcome'] = df2['poutcome'].astype('category').cat.codes
         df2['month'] = df2['month'].astype('category').cat.codes
         df2['default'] = df2['default'].astype('category').cat.codes
         df2['loan'] = df2['loan'].astype('category').cat.codes
         df2['housing'] = df2['housing'].astype('category').cat.codes

         target_name = 'deposit'
         X = df2.drop('deposit', axis=1)

         label=df2[target_name]

         X_train, X_test, y_train, y_test = train_test_split(X,label,test_size=0.2, random_state=42, str

         model = XGBClassifier(learning_rate = 0.5, n_estimators=300, max_depth=5)
         model.fit(X_train, y_train)

         importances = model.feature_importances_
         feature_names = df2.drop('deposit', axis=1).columns
         indices = np.argsort(importances)[::-1]
```



```
# Print the feature ranking
print("Feature ranking:")

for f in range(X_train.shape[1]):
    print("%d. feature %d (%f)" % (f + 1, indices[f], importances[indices[f]]))

def feature_importance_graph(indices, importances, feature_names):
    plt.figure(figsize=(12,6))
    plt.title("Determining Feature importances \n with XGBoost", fontsize=18)
    plt.barh(range(len(indices)), importances[indices], color='#ffd700',  align="center")
    plt.yticks(range(len(indices)), feature_names[indices], rotation='horizontal',fontsize=14)
    plt.ylim([-1, len(indices)])

feature_importance_graph(indices, importances, feature_names)
plt.show()
```


Determining Feature importances
with XGBoost Classifier
default
job
previous
balance
marital
campaign
day
age
education
loan
pdays
month
duration
housing
poutcome
contact
0.000
0.025
0.050
0.075
0.100
0.125
0.150
0.175
0.200