Data Mining Project

تعتبر بيانات "telco_customer_churn" مجموعة بيانات متعلقة بشركة اتصالات تحتوي على معلومات حول عملائها وتحويلهم (الانتقال) إلى منافسين آخرين. تم جمع هذه البيانات لدراسة العوامل التي قد تؤدي إلى انتقال العملاء إلى شركات منافسة، ومن ثم تحسين تجربة العملاء والحفاظ عليهم.

تحتوي مجموعة البيانات على معلومات حول العملاء مثل الجنس والعمر والحالة الاجتماعية، بالإضافة إلى معلومات الاشتراك والدفع والاستخدام. كما تحتوي المجموعة على معلومات حول الخدمات التي يستخدمها العملاء، مثل الإنترنت والهاتف والتلفزيون.

تتضمن المجموعة أيضاً معلومات حول مدة الاشتراك لدى العملاء وما إذا كانوا قد أنهوا اشتراكهم في الخدمة أو لا. ويتم تمثيل هذه المعلومات في شكل ميزة "churn" التي تشير إلى ما إذا كان العميل قد قام بإلغاء اشتراكه أو لا.

في القسم الأول تم العمل على تنظيف الداتا المستخدمة

والتعرف على الداتا بشكل جيد.

في البداية تم استدعاء المكتبات المستخدمة في المشروع

في قسم **Exploratory Data Analysis**

تحميل الداتا باستخدام pandas وطباعة عدد العينات والتي بلغت 7043 .

استخدام الدالة info الموجودة في pandas لمعرفة feature الموجودة نوع كل feature وكم عدد القيم null لكل feature .
حيث تمثل

1. CustomerID: رقم معرف فريد لكل عميل في شركة الاتصالات.

2. Gender: الجنس (ذكر/أنثى) للعميل.

3. SeniorCitizen: متغير ثنائي يشير إلى ما إذا كان العميل مسنًا (بعمر ٦٥ عامًا فأكثر) أم لا .

4. Partner: متغير ثنائي يشير إلى ما إذا كان العميل متزوجًا أم لا.

5. Dependents: متغير ثنائي يشير إلى ما إذا كان لدى العميل أفراد يعتمدون عليه ماديًا أم لا.

6. Tenure: عدد الأشهر التي قضاها العميل في الاشتراك مع شركة الاتصالات.

7. PhoneService: متغير ثنائي يشير إلى ما إذا كان العميل يستخدم خدمة الهاتف الثابت أم لا.

8. MultipleLines: متغير ثنائي يشير إلى ما إذا كان العميل يستخدم خدمة الهاتف الثابت مع خطوط متعددة أم لا.

9. InternetService: نوع خدمة الإنترنت التي يستخدمها العميل (DSL / Fiber optic / No).

10. OnlineSecurity: متغير ثنائي يشير إلى ما إذا كان لدى العميل خدمة حماية الإنترنت الخاصة به أم لا.

11. OnlineBackup: متغير ثنائي يشير إلى ما إذا كان لدى العميل خدمة النسخ الاحتياطي للإنترنت الخاصة به أم لا.

12. DeviceProtection: متغير ثنائي يشير إلى ما إذا كان لدى العميل خدمة حماية الأجهزة الخاصة به أم لا.

13. TechSupport: متغير ثنائي يشير إلى ما إذا كان لدى العميل خدمة الدعم التقني الخاصة به أم لا.

14. StreamingTV: متغير ثنائي يشير إلى ما إذا كان العميل يستخدم خدمة تدفق التلفزيون أم لا.

15. StreamingMovies: متغير ثنائي يشير إلى ما إذا كان العميل يستخدم خدمة تدفق الأفلام أم لا.

16. Contract: نوع العقد الذي يستخدمه العميل (Month-to-month, One year, Two year).

17. PaperlessBilling: متغير ثنائي يشير إلى ما إذا كان العميل يستخدم فواتير إلكترونية أم فواتير ورقية.

18. PaymentMethod: طريقة الدفع التي يستخدمها العميل (Electronic check, Mailed check, Bank transfer (automatic), Credit card (automatic)).

19. MonthlyCharges: المبلغ الذي يدفعه العميل شهريًا لشركة الاتصالات.

20. TotalCharges: المبلغ الإجمالي الذي دفعه العميل لشركة الاتصالات منذ بدء الاشتراك.

21. Churn: متغير ثنائي يشير إلى ما إذا كان العميل قد قام بإلغاء الاشتراك أم لا.

في قسم **Data Preparation**

بعد استخدام info نلاحظ انو TotalCharges يجب أن تكون float وليس object سوف نقوم بتحويل نوعها باستخدام to_numeric .

استخدام info مرة ثانية تشير إلى وجود 7032 قيمة غير فارغة في TotalCharges وبالتالي يوجد 2 قيمة فارغة بالتالي سوف نقوم ملء الفراغات بقيمة 0.

تم تحويل قيم الاعمدة واسماءها إلى احرف صغيره لسهولة التعامل معها.

تم تحويل عمود Churn إلى أرقام بدلا من سلسلة نصية حيث

yes → 1, no → 0

تم طباعة القيم التي تمثل كل صنف فيوجد 5174 الذين قاموا بإلغاء الاشتراك و 1869 لم يقوموا بإلغاء الاشتراك هذا يعني ان التصنيف هنا هو تصنيف غير متوازن imbalanced classification

وهذه المعلومة تعني أن مقياس accuracy لن يكون مفيد في تقيم النماذج لاحقا لان هذا المقياس يكون افضل في حال كان عدد العينات للصنفين تقريبا متساوي.

تقسيم الداتا إلى 70% للتدريب و 30% للاختبار باستخدام train_test_split من مكتبة sklearn وطباعة عدد العينات للتدريب والاختبار

في قسم **One-hot encoding**

تم تحويل كل سطر في الداتا إلى dict لكي تناسب عملية الترميز المتقدمة.

تم استدعاء DictVectorizer من مكتبة sklearn حيث يعتبر من الطرائق المتقدمة في عملية الترميز حيث يقوم بتحويل المتغيرات في كل سطر إلى متجه وبشكل تلقائي عندما تكون القيمة numerical فأنه يأخذها كما هي في حين عند وجود قيمة نصيه في categorical يعطيها رقم متقطع بحسب عدد categorical الكلي.

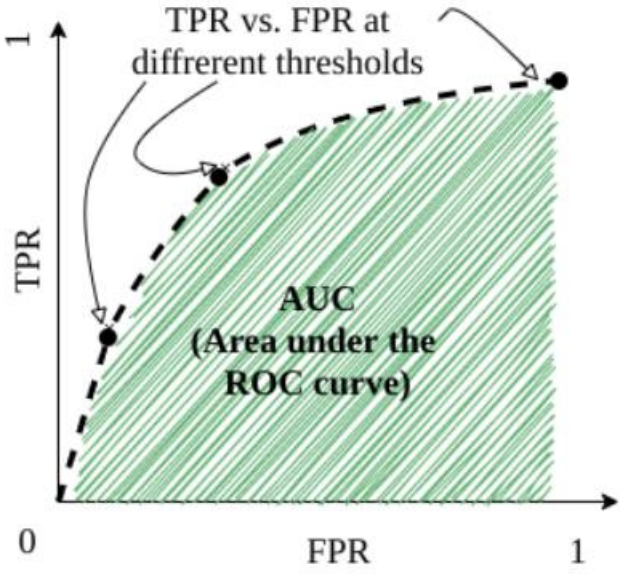
في القسم الثاني تم تدريب أكثر من model وتقييم كل موديل باستخدام AUC ,ROC Curve وبعد ذلك اختيار model الذي يحقق اكبر AUC

ROC Curve هو منحني بياني يمثل قدرة النموذج على التمييز بين فئتين مثل فئة المريض والشخص السليم ويتم ذلك عن طريق حساب threshold بين معدل إلايجابيات الحقيقية TPR ومعدل الايجابيات الزائفة TPR حيث

TPR هو عدد الحالات الايجابية الحقيقة في المجموعة أي الحالات التي تمكن model من تصنيفها بشكل صحيح و هي إيجابية وتحسب بالقانون TPR = TP / (TP + FN)

FPR هو عدد الحالات الايجابية الزائفة في المجموعة أي الحالات التي تمكن model من تصنيفها بشكل صحيح و هي سلبية وتحسب بالقانون TPR = FP / (FP + FN)

يمكن اختصار الكلام السابق في المخطط البياني التالي



حيث يمثل الخط البياني roc مقياس لقابلية الفصل بين الفئات و المساحة التي تحت curve وتدعى auc حيث تخبرنا إلى إي مدى يمكن ل model التمييز بين الفئتين وكلما كانت هذا المساحة أكبر فهذا يعطي فكرة أن ال model أفضل في التنبؤ.

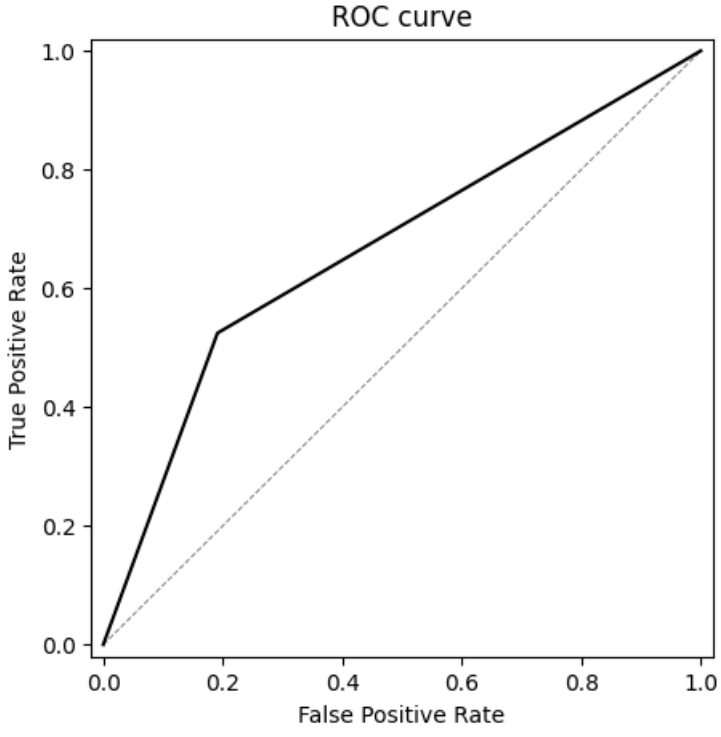
في قسم **DecisionTreeClassifier**

تم استدعاء DecisionTreeClassifier من مكتبة sklearn

تم اخذ instance من DecisionTreeClassifier وبدء عملية التدريب ثم إيجاد النتائج لداتا الاختبار باستخدام model المدرب

تم حساب قيمة auc التي تحدثنا عنها سابقا باستخدام roc_auc_score الموجودة في sklearn.metrics

تم رسم المخطط التي تحدثنا عنه سابقا بأخذ tpr, fpr من roc_curve وتم استخدام matplotlib في عملية الرسم



في قسم **Naive Bayes**

تم استدعاء Naive Bayes من مكتبة sklearn

تم اخذ instance من Naive Bayes وبدء عملية التدريب ثم إيجاد

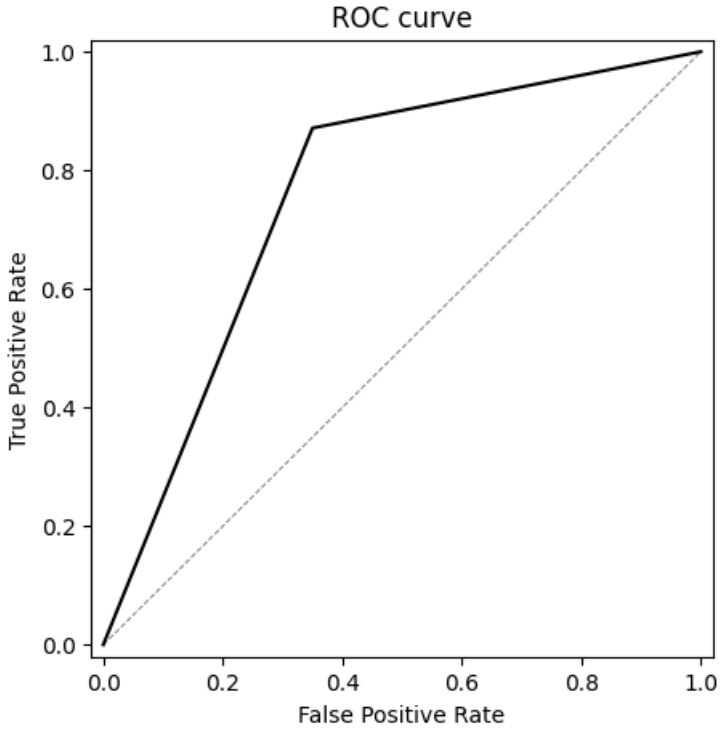
النتائج لداتا الاختبار باستخدام model المدرب

تم حساب قيمة auc التي تحدثنا عنها سابقا باستخدام

roc_auc_score الموجودة في sklearn.metrics

تم رسم المخطط التي تحدثنا عنه سابقا بأخذ tpr, fpr من roc_curve

وتم استخدام matplotlib في عملية الرسم



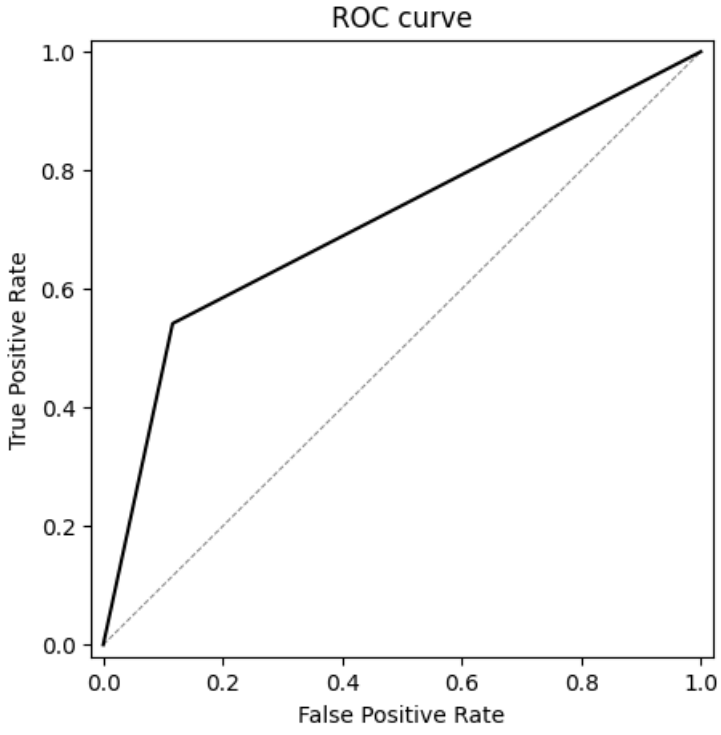
في قسم **RandomForest**

تم استدعاء RandomForest من مكتبة sklearn

تم اخذ instance من RandomForest وبدء عملية التدريب ثم إيجاد النتائج لداتا الاختبار باستخدام model المدرب

تم حساب قيمة auc التي تحدثنا عنها سابقا باستخدام roc_auc_score الموجودة في sklearn.metrics

تم رسم المخطط التي تحدثنا عنه سابقا بأخذ tpr, fpr من roc_curve وتم استخدام matplotlib في عملية الرسم



في قسم **XGBoost**

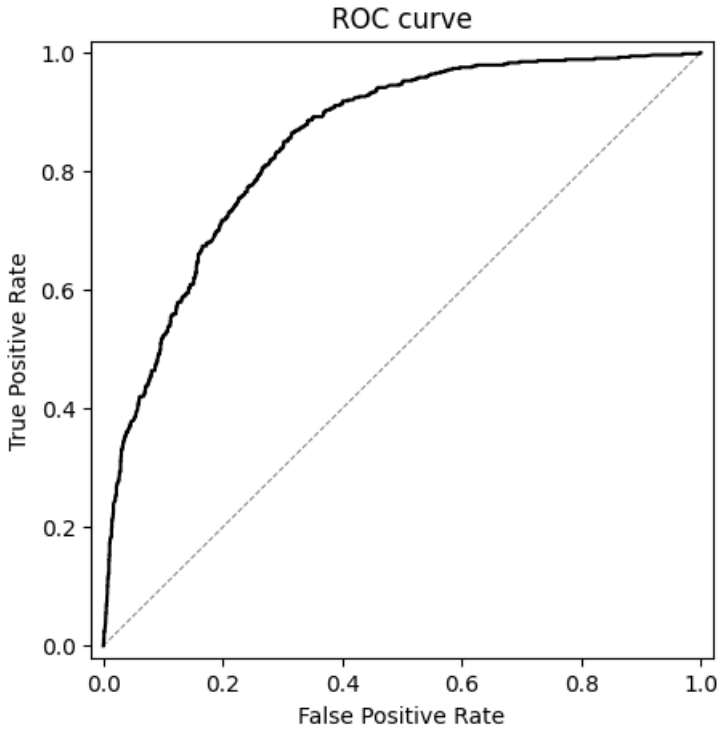
تم استخدام DMatrix من مكتبة XGBoost لتجهيز البيانات للتعامل مع XGBoost وتم تطبيق ذلك على التدريب والاختبار

تم اخذ xgb_params وهي dict لوضع معلمات التدريب مثل معدل التعلم eta وعمق شجرة القرار max_depth وعدد الثريدات لكي نستفاد من التفرعية على cpu .

بدء عملية التدريب ثم إيجاد النتائج لداتا الاختبار باستخدام model المدرب

تم حساب قيمة auc التي تحدثنا عنها سابقا باستخدام roc_auc_score الموجودة في sklearn.metrics

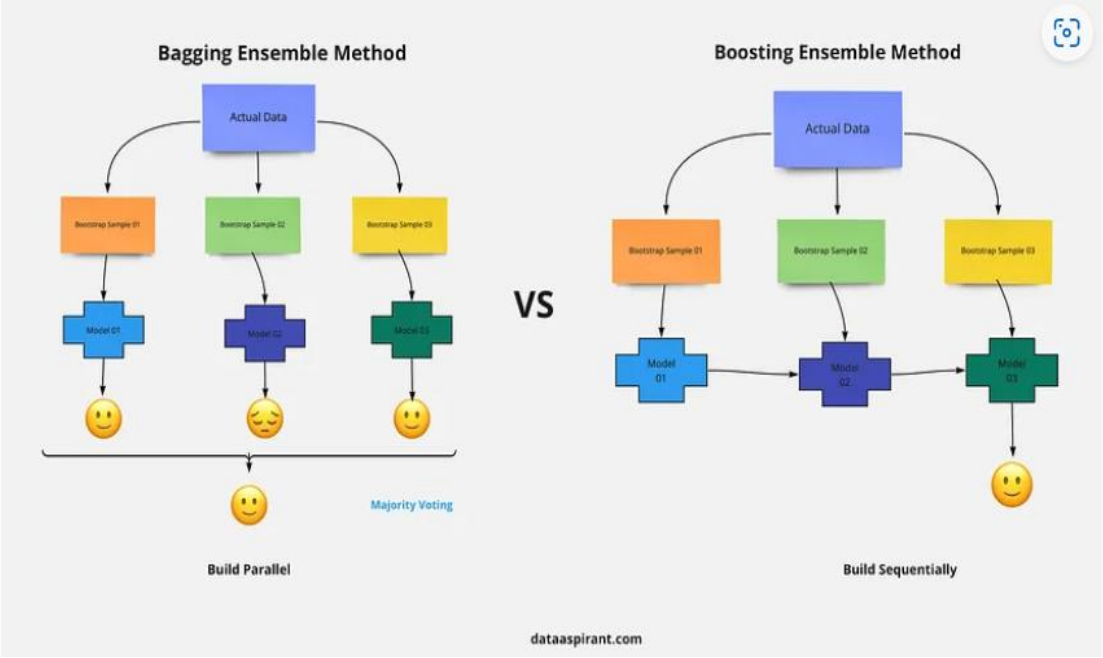
تم رسم المخطط التي تحدثنا عنه سابقا بأخذ tpr, fpr من roc_curve وتم استخدام matplotlib في عملية الرسم.



Model	AUC
Decision Tree Classifier	0.66
Naïve Bayes	0.761
Random Forest Classifier	0.706
XGBoost	0.849

طبعا من خلال الجدول السابق الذي يوضح كل model وقيمة auc الخاصة به فأن افضل model هو xgboost

فقرة إضافية لفهم XGBoost و RandomForest استناد على الصورة التالية



يعتبر RandomForest من نماذج Bagging حيث يتم في نماذج Bagging أخذ مجموعة من Decision Trees ويتم تدريب كل شجرة قرار على التوازي وعلى عينات عشوائية مختلفة من البيانات وبعد نهاية عملية التدريب وتجريب عينة على model تقوم كل شجرة قرار بتوقع نتيجة العينة ومن ثم يتم أخذ النتيجة التي تحقق اعلى تصويت بين اشجار القرار

في حين يعتبر XGBoost من نماذج Boosting حيث يتم في نماذج Boosting أخذ مجموعة من Decision Trees ويتم تدريب كل شجرة قرار على التسلسل بناء على الخطأ الذي تم الحصول عليه من الشجرة السابقة وتحاول كل شجرة تصحيح هذا الخطأ وتحسين الدقة العامة لل model وبعد نهاية عملية التدريب وتجريب عينة على model يتم التوقع بناء على الاوزان النهائية للميزات و التي وصل إليها model خلال عملية التدريب.

إعداد: عمار معلا